About this Book

Contents

Syllabus

- About
- Tools and Resources
- Data Science Achievements
- Grading
- Grading Policies
- Course Style Guide
- Support
- General URI Policies
- Communications & Office Hours

Notes

- 1. Welcome & What is Data Science
- 2. Iterables and Pandas Data Frames
- 3. DataFrames from other sources
- 4. Exploratory Data Analysis

Assignments

- 1. Assignment 1: Setup, Syllabus, and Review
- 2. Assignment 2: Practicing Python and Accessing Data
- 3. Assignment 3: Exploratory Data Analysis

Portfolio

- Portfolio
- Formatting Tips
- Portfolio Check 1 Ideas
- Check 2 Ideas
- Check 3 Ideas

FAQ

- FAQ
- Syllabus and Grading FAQ
- Git and GitHub
- Code Errors

Resources

Skip to main content

- References on Python
- Cheatsheet
- Data Sources
- General Tips and Resources
- How to Study in this class
- Getting Help with Programming
- Terminals and Environments
- Getting Organized for class
- Advice from FA2020 Students
- Advice from FA2021 Students

Welcome to the course manual for CSC310 at URI with Professor Brown.

This class meets TTh 5-6:15pm in Engineering 040.

This website will contain the syllabus, class notes, and other reference material for the class.

Land University of Rhode Island Land Acknowledgment

Note

The University of Rhode Island land acknowledgment is a statement written by members of the University community in close partnership with members of the Narragansett Tribe. The statement recognizes and pays tribute to the people who lived on and stewarded the land on which the University now resides. The statement seeks to show gratitude and respect to Indigenous people and cultures and build community with the Narragansett Nation and other Native American tribes.

The University of Rhode Island occupies the traditional stomping ground of the Narragansett Nation and the Niantic People. We honor and respect the enduring and continuing relationship between the Indigenous people and this land by teaching and learning more about their history and present-day communities, and by becoming stewards of the land we, too, inhabit.

Navigating the Sections

The Syllabus section has logistical operations for the course broken down into sections. You can also read straight through by starting in the first one and navigating to the next section using the arrow navigation at the end of the page.

This site is a resource for the course. We do not follow a text book for this course, but all notes from class are posted in the notes section, accessible on the left hand side menu, visible on large screens and in the menu on mobile.

The resources section has links and short posts that provide more context and explanation. Content in this section is for the most part not strictly the material that you'll be graded on, but it is often material that will help you understand and grow as a programmer and data scientist.

Reading each page

Skip to main content

All class notes can be downloaded in multiple formats, including as a notebook. Some pages of the syllabus and resources are also notebooks, if you want to see behind the curtain of how I manage the course information.

Try it Yourself

Notes will have exercises marked like this

Question from Class

Questions that are asked in class, but unanswered at that time will be answered in the notes and marked with a box like this. Long answers will be in the main notes

Further reading

Notes that are mostly links to background and context will be highlighted like this. These are optional, but will mostly help you understand code excerpts they relate to.

Hint

Both notes and assignment pages will have hints from time to time. Pay attention to these on the notes, they'll typically relate to things that will appear in the assignment.

Think Ahead

Think ahead boxes will guide you to start thinking about what can go into your portfolio to build on the material at hand.

Click here!

Special tips will be formatted like this

Check your Comprehension

Questions to use to check your comprehension will looklike this

About

About the topic

Data science exists at the intersection of computer science, statistics, and domain expertise. That means writing programs to access and manipulate data so that it becomes available for analysis using statistical and machine learning techniques is at the core of data science. Data scientists use their data and analytical ability to find and interpret rich data sources; manage large amounts of data despite hardware, software, and bandwidth constraints; merge data sources; ensure consistency of datasets; create visualizations to aid in understanding data; build mathematical models using the data; and present and communicate the data insights/findings.

Skip to main content

About the goals and preparation

This course provides a survey of data science. Topics include data driven programming in Python; data sets, file formats and meta-data; descriptive statistics, data visualization, and foundations of predictive data modeling and machine learning; accessing web data and databases; distributed data management. You will work on weekly programming problems such as accessing data in database and visualize it or build machine learning models of a given data set.

Basic programming skills (CSC201 or CSC211) are a prerequisite to this course. This course is a prerequisite course to machine learning, where you learn how machine learning algorithms work. In this course, we will start with a very fast review of basic programming ideas, since you've already done that before. We will learn how to *use* machine learning algorithms to do data science, but not how to *build* machine learning algorithms, we'll use packages that implement the algorithms for us.

About the course

This course is designed to make you a better programmer while learning data science. You may be stronger in one of those areas than the other at the beginning, but you should grow in both areas by the end of the semester.

About this syllabus

This syllabus is a *living* document and accessible from BrightSpace, as a pdf for download directly, and online at rhodyprog4ds.github.io/BrownFall23/syllabus. If you choose to download a copy of it, note that it is only a copy. You can get notification of changes from GitHub by "watching" the repository. You can view the date of changes and exactly what changes were made on the Github commits page.

Creating an issue on the repository is also a good way to ask questions about anything in the course it will prompt additions and expand the FAQ section.

About your instructor

Name: Dr. Sarah Brown Office hours: TBA via zoom, link on GitHub Org Page

Dr. Brown is an Assistant Professor of Computer Science, who does research on how social context changes machine learning. Dr. Brown earned a PhD in Electrical Engineering from Northeastern University, completed a postdoctoral fellowship at University of California Berkeley, and worked as a postdoctoral research associate at Brown University before joining URI. At Brown University, Dr. Brown taught the Data and Society course for the Master's in Data Science Program.

Important

For assignment or notes specific issues, a comment on the corresponding repository is the best. I cannot help you with code issues from screenshots.

Not

Whet
not m

The best way to contact me for general questions is e-mail or by dropping into my office hours. Please include `[CSC310]` or `[DSP310]` in the subject line of your email along with the topic of your message. This is important, because your messages are important, but I also get a lot of e-mail. Consider these a cheat code to my inbox: I have setup a filter that will flag your e-mail if you use one of those in the subject to ensure that I see it. I rarely check e-mail between 6pm and 9am, on weekends or holidays. You might see me post or send things during these hours, but I will not reliably see emails that arrive during those hours.

Skip to main content

Tools and Resources

We will use a variety of tools to conduct class and to facilitate your programming. You will need a computer with Linux, MacOS, or Windows. It is unlikely that a tablet will be able to do all of the things required in this course. A Chromebook may work, especially with developer tools turned on. Ask Dr. Brown if you need help getting access to an adequate computer.

All of the tools and resources below are either:

- paid for by URI **OR**
- freely available online.

BrightSpace

This will be the central location from which you can access links to other materials. Any links that are for private discussion among those enrolled in the course will be available only from our course Brightspace site.

Prismia chat

Our class link for Prismia chat is available on Brightspace. We will use this for chatting and in-class understanding checks.

On Prismia, all students see the instructor's messages, but only the Instructor and TA see student responses.

Course website

The course manual will have content including the class policies, scheduling, class notes, assignment information, and additional resources. This will be linked from Brightspace and available publicly online at rhodyprog4ds.github.io/BrownSpring23/. Links to the course reference text and code documentation will also be included here in the assignments and class notes.

GitHub

You will need a GitHub Account. If you do not already have one, please create one by the first day of class. If you have one, but have not used it recently, you may need to update your password and login credentials as the Authentication rules changed over the summer. In order to use the command line with https, you will need to us the GitHub CLI or create a Personal Access Token for each device you use. In order to use the command line with SSH, set up your public key.

Programming Environment

This a programming course, so you will need a programming environment. In order to complete assignments you need the items listed in the requirements list. The easiest way to meet these requirements is to follow the recommendations below. I will provide instruction assuming that you have followed the recommendations.

Requirements:

- Python with scientific computing packages (numpy, scipy, jupyter, pandas, seaborn, sklearn)

Skip to main content

- A web browser compatible with Jupyter Notebooks

> **Warning**
>
> Everything in this class will be tested with the up to date (or otherwise specified) version of Jupyter Notebooks. Google Colab is similar, but not the same, and some things may not work there. It is an okay backup, but should not be your primary work environment.

> **Not**
>
> all Gi
> instru
> interf
> instru
> may
> built i
> team

Recommendation:

- Install python via Anaconda
- if you use Windows, install Git with GitBash (video instructions).
- if you use MacOS, install Git with the Xcode Command Line Tools. On Mavericks (10.9) or above you can do this by trying to run git from the Terminal the very first time. `git --version`
- if you use Chrome OS, follow these instructions:

1. Find Linux (Beta) in your settings and turn that on.
2. Once the download finishes a Linux terminal will open, then enter the commands: sudo apt-get update and sudo apt-get upgrade. These commands will ensure you are up to date.
3. Install tmux with:

```
sudo apt -t stretch-backports install tmux
```

4. Next you will install nodejs, to do this, use the following commands:

```
curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash
sudo apt-get install -y nodejs
sudo apt-get install -y build-essential.
```

5. Next install Anaconda's Python from the website provided by the instructor and use the top download link under the Linux options.
6. You will then see a .sh file in your downloads, move this into your Linux files.
7. Make sure you are in your home directory (something like home/YOURUSERNAME), do this by using the `pwd` command.
8. Use the `bash` command followed by the file name of the installer you just downloaded to start the installation.
9. Next you will add Anaconda to your Linux PATH, do this by using the `vim .bashrc` command to enter the .bashrc file, then add the `export PATH=/home/YOURUSERNAME/anaconda3/bin/:$PATH` line. This can be placed at the end of the file.
10. Once that is inserted you may close and save the file, to do this hold escape and type `:x`, then press enter. After doing that you will be returned to the terminal where you will then type the source .bashrc command.
11. Next, use the `jupyter notebook –generate-config` command to generate a Jupyter Notebook.
12. Then just type `jupyter lab` and a Jupyter Notebook should open up.

Optional:

- Text Editor: you may want a text editor outside of the Jupyter environment. Jupyter can edit markdown files (that you'll need for your portfolio), in browser, but it is more common to use a text editor like Atom or Sublime for this purpose.

Skip to main content

- Windows
- Mac

On Mac, to install python via environment, this article may be helpful

- I don't have a video for linux, but it's a little more straight forward.

Textbook

The text for this class is a reference book and will not be a source of assignments. It will be a helpful reference and you may be directed there for answers to questions or alternate explanations ot topics.

Python for Data Science is available free online:

Zoom (backup and office hours only)

This is where we will meet if for any reason we cannot be in person. You will find the link to class zoom sessions on Brightspace.

URI provides all faculty, staff, and students with a paid Zoom account. It *can* run in your browser or on a mobile device, but you will be able to participate in class best if you download the Zoom client on your computer. Please log in and configure your account. Please add a photo of yourself to your account so that we can still see your likeness in some form when your camera is off. You may also wish to use a virtual background and you are welcome to do so.

Class will be interactive, so if you cannot be in a quiet place at class time, headphones with a built in microphone are strongly recommended.

For help, you can access the instructions provided by IT.

---

[1] Too long; didn't read.

Data Science Achievements

In this course there are 5 learning outcomes that I expect you to achieve by the end of the semester. To get there, you'll focus on 15 smaller achievements that will be the basis of your grade. This section will describe how the topics covered, the learning outcomes, and the achievements are covered over time. In the next section, you'll see how these achievements turn into grades.

Learning Outcomes

By the end of the semester

1. (process) Describe the process of data science, define each phase, and identify standard tools
2. (data) Access and combine data in multiple formats for analysis
3. (exploratory) Perform exploratory data analyses including descriptive statistics and visualization
4. (modeling) Select models for data by applying and evaluating mutiple models to a single dataset
5. (communicate) Communicate solutions to problems with data in common industry formats

We will build your skill in the `process` and `communicate` outcomes over the whole semester. The middle three skills will correspond roughly to the content taught for each of the first three portfolio checks.

Schedule

The course will meet in . Every class will include participatory live coding (instructor types code while explaining, students follow along) instruction and small exercises for you to progress toward level 1 achievements of the new skills introduced in class that day.

Each Assignment will have a deadline posted on the assignment page, typically the same day each week. Portfolio deadlines will be announced at least 2 weeks in advance.

week	topics	skills
1	[admin, python review]	process
2	Loading data, Python review	[access, prepare, summarize]
3	Exploratory Data Analysis	[summarize, visualize]
4	Data Cleaning	[prepare, summarize, visualize]
5	Databases, Merging DataFrames	[access, construct, summarize]
6	Modeling, classification performance metrics, cross validation	[evaluate]
7	Naive Bayes, decision trees	[classification, evaluate]
8	Regression	[regression, evaluate]
9	Clustering	[clustering, evaluate]
10	SVM, parameter tuning	[optimize, tools]
11	KNN, Model comparison	[compare, tools]
12	Text Analysis	[unstructured]
13	Images Analysis	[unstructured, tools]
14	Deep Learning	[tools, compare]

Achievement Definitions

The table below describes how your participation, assignments, and portfolios will be assessed to earn each achievement. The keyword for each skill is a short name that will be used to refer to skills throughout the course materials; the full description of the skill is in this table.

keyword	skill	Level 1	Level 2	Level 3
python	pythonic code writing	python code that mostly runs, occasional pep8 adherance	python code that reliably runs, frequent pep8 adherance	reliable, efficient, pythonic code that consistently adheres to pep8
process	describe data science as a process	Identify basic components of data science	Describe and define each stage of the data science process	Compare different ways that data science can facilitate decision making
access	access data in multiple formats	load data from at least one format; identify the most common data formats	Load data for processing from the most common formats; Compare and constrast most common formats	access data from both common and uncommon formats and identify best practices for formats in different contexts
construct	construct datasets from multiple sources	identify what should happen to merge datasets or when they can be merged	apply basic merges	merge data that is not automatically aligned
summarize	Summarize and describe data	Describe the shape and structure of a dataset in basic terms	compute summary statndard statistics of a whole dataset and grouped data	Compute and interpret various summary statistics of subsets of data
visualize	Visualize data	identify plot types, generate basic plots from pandas	generate multiple plot types with complete labeling with pandas and seaborn	generate complex plots with pandas and plotting libraries and customize with matplotlib or additional parameters
prepare	prepare data for analysis	identify if data is or is not ready for analysis, potential problems with data	apply data reshaping, cleaning, and filtering as directed	apply data reshaping, cleaning, and filtering manipulations reliably and correctly by assessing data as received
evaluate	Evaluate model performance	Explain and compute basic performance metrics for different data science tasks	Apply and interpret basic model evaluation metrics to a held out test set	Evaluate a model with multiple metrics and cross validation
classification	Apply classification	identify and describe what classification is, apply pre-fit classification models	fit, apply, and interpret preselected classification model to a dataset	fit and apply classification models and select appropriate classification models for different contexts
regression	Apply Regression	identify what data that can be used for regression looks like	fit and interpret linear regression models	fit and explain regrularized or nonlinear regression
clustering	Clustering	describe what clustering is	apply basic clustering	apply multiple clustering techniques, and interpret results
optimize	Optimize model parameters	Identify when model parameters need to be optimized	Optimize basic model parameters such as model order	Select optimal parameters based of mutiple quanttiateve criteria and automate parameter tuning

keyword	skill	Level 1	Level 2	Level 3
compare	compare models	Qualitatively compare model classes	Compare model classes in specific terms and fit models in terms of traditional model performance metrics	Evaluate tradeoffs between different model comparison types
representation	Choose representations and transform data	Identify options for representing text and categorical data in many contexts	Apply at least one representation to transform unstructured or inappropriately data for model fitting or summarizing	apply transformations in different contexts OR compare and contrast multiple representations a single type of data in terms of model performance
workflow	use industry standard data science tools and workflows to solve data science problems	Solve well strucutred fully specified problems with a single tool pipeline	Solve well-strucutred, open-ended problems, apply common structure to learn new features of standard tools	Independently scope and solve realistic data science problems OR independently learn releated tools and describe strengths and weakensses of common tools

Assignments and Skills

Using the keywords from the table above, this table shows which assignments you will be able to demonstrate which skills and the total number of assignments that assess each skill. This is the number of opportunities you have to earn Level 2 and still preserve 2 chances to earn Level 3 for each skill.

keyword	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	# Assignments
python	1	1	0	1	1	0	0	0	0	0	0	0	0	4
process	1	0	0	0	0	1	1	1	1	1	1	0	0	7
access	0	1	1	1	1	0	0	0	0	0	0	0	0	4
construct	0	0	0	0	1	0	1	1	0	0	0	0	0	3
summarize	0	0	1	1	1	1	1	1	1	1	1	1	1	11
visualize	0	0	1	1	0	1	1	1	1	1	1	1	1	10
prepare	0	0	0	1	1	0	0	0	0	0	0	0	0	2
evaluate	0	0	0	0	0	1	1	1	0	1	1	0	0	5
classification	0	0	0	0	0	0	1	0	0	1	0	0	0	2
regression	0	0	0	0	0	0	0	1	0	0	1	0	0	2
clustering	0	0	0	0	0	0	0	0	1	0	1	0	0	2
optimize	0	0	0	0	0	0	0	0	0	1	1	0	0	2
compare	0	0	0	0	0	0	0	0	0	0	1	0	1	2
representation	0	0	0	0	0	0	0	0	0	0	0	1	1	2
workflow	0	0	0	0	0	0	0	0	0	1	1	1	1	4

Skip to main content

> **Warning**
>
> **process** achievements are accumulated a little slower. Prior to portfolio check 1, only level 1 can be earned. Portfolio check 1 is the first chance to earn level 2 for process, then level 3 can be earned on portfolio check 2 or later.

Portfolios and Skills

The objective of your portfolio submissions is to earn Level 3 achievements. The following table shows what Level 3 looks like for each skill and identifies which portfolio submissions you can earn that Level 3 in that skill.

keyword	Level 3	P1	P2	P3	P4
python	reliable, efficient, pythonic code that consistently adheres to pep8	1	1	0	1
process	Compare different ways that data science can facilitate decision making	0	1	1	1
access	access data from both common and uncommon formats and identify best practices for formats in different contexts	1	1	0	1
construct	merge data that is not automatically aligned	1	1	0	1
summarize	Compute and interpret various summary statistics of subsets of data	1	1	0	1
visualize	generate complex plots with pandas and plotting libraries and customize with matplotlib or additional parameters	1	1	0	1
prepare	apply data reshaping, cleaning, and filtering manipulations reliably and correctly by assessing data as received	1	1	0	1
evaluate	Evaluate a model with multiple metrics and cross validation	0	1	1	1
classification	fit and apply classification models and select appropriate classification models for different contexts	0	1	1	1
regression	fit and explain regrularized or nonlinear regression	0	1	1	1
clustering	apply multiple clustering techniques, and interpret results	0	1	1	1
optimize	Select optimal parameters based of mutiple quanttiateve criteria and automate parameter tuning	0	0	1	1
compare	Evaluate tradeoffs between different model comparison types	0	0	1	1
representation	apply transformations in different contexts OR compare and contrast multiple representations a single type of data in terms of model performance	0	0	1	1
workflow	Independently scope and solve realistic data science problems OR independently learn releated tools and describe strengths and weakensses of common tools	0	0	1	1

Detailed Checklists

python-level1

python code that mostly runs, occasional pep8 adherance

- [ ] use of control structures
- [ ] callable functions

Skip to main content

- [ ] correct use of variables
- [ ] use of logical operators

python-level2

python code that reliably runs, frequent pep8 adherance

- [ ] descriptive variable names
- [ ] pythonic loops
- [ ] effective use of return vs side effects in functions
- [ ] correct, effective use of builtin python iterable types (lists & dictionaries)

python-level3

reliable, efficient, pythonic code that consistently adheres to pep8

- [ ] pep8 adherant variable, file, class, and function names
- [ ] effective use of multi-paradigm abilities for efficiency gains
- [ ] easy to read code that adheres to readability over other rules

process-level1

Identify basic components of data science

- [ ] identify component disciplines
- [ ] idenitfy phases

process-level2

Describe and define each stage of the data science process

- [ ] correctly defines stages
- [ ] identifies stages in use
- [ ] describes general goals as well as a specific processes

process-level3

Compare different ways that data science can facilitate decision making

- [ ] describes exceptions to process and iteration in process
- [ ] connects choices at one phase to impacts in other phases
- [ ] connects data science steps to real world decisions

access-level1

Skip to main content

- [ ] use at least one pandas `read_` function correctly
- [ ] name common types
- [ ] describe the structure of common types

access-level2

Load data for processing from the most common formats; Compare and constrast most common formats

- [ ] load data from at least two of (.csv, .tsv, .dat, database, .json)
- [ ] describe advantages and disadvantages of most commone types
- [ ] descive how most common types are different

access-level3

access data from both common and uncommon formats and identify best practices for formats in different contexts

- [ ] load data from at least 1 uncommon format
- [ ] describe when one format is better than another

construct-level1

identify what should happen to merge datasets or when they can be merged

- [ ] identify what the structure of a merged dataset should be (size, shape, columns)
- [ ] idenitfy when datasets can or cannot be merged

construct-level2

apply basic merges

- [ ] use 3 different types of merges
- [ ] choose the right type of merge for realistic scenarios

construct-level3

merge data that is not automatically aligned

- [ ] manipulate data to make it mergable
- [ ] identify how to combine data from many sources to answer a question
- [ ] implement stesp to combine data from multiple sources

summarize-level1

Describe the shape and structure of a dataset in basic terms

- [ ] use attributes to produce a description of a dataset

Skip to main content

summarize-level2

compute and interpret summary standard statistics of a whole dataset and grouped data

- [ ] compute descriptive statistics on whole datasets
- [ ] apply individual statistics to datasets
- [ ] group data by a categorical variable for analysis
- [ ] apply split-apply-combine paradigm to analyze data
- [ ] interprete statistics on whole datasets
- [ ] interpret statistics on subsets of data

summarize-level3

Compute and interpret various summary statistics of subsets of data

- [ ] produce custom aggregation tables to summarize datasets
- [ ] compute multivariate summary statistics by grouping
- [ ] compute custom cacluations on datasets

visualize-level1

identify plot types, generate basic plots from pandas

- [ ] generate at least two types of plots with pandas
- [ ] identify plot types by name
- [ ] interpret basic information from plots

visualize-level2

generate multiple plot types with complete labeling with pandas and seaborn

- [ ] generate at least 3 types of plots
- [ ] use correct, complete, legible labeling on plots
- [ ] plot using both pandas and seaborn
- [ ] interpret multiple types of plots to draw conclusions

visualize-level3

generate complex plots with pandas and plotting libraries and customize with matplotlib or additional parameters

- [ ] use at least two libraries to plot
- [ ] generate figures with subplots
- [ ] customize the display of a plot to be publication ready
- [ ] interpret plot types and explain them for novices
- [ ] choose appopriate plot types to convey information

Skip to main content

prepare-level1

identify if data is or is not ready for analysis, potential problems with data

- [ ] identify problems in a dataset
- [ ] anticipate how potential data setups will interfere with analysis
- [ ] describe the structure of tidy data
- [ ] label data as tidy or not

prepare-level2

apply data reshaping, cleaning, and filtering as directed

- [ ] reshape data to be analyzable as directed
- [ ] filter data as directed
- [ ] rename columns as directed
- [ ] rename values to make data more analyzable
- [ ] handle missing values in at least two ways
- [ ] transform data to tidy format

prepare-level3

apply data reshaping, cleaning, and filtering manipulations reliably and correctly by assessing data as received

- [ ] identify issues in a dataset and correctly implement solutions
- [ ] convert varialbe representation by changing types
- [ ] change variable representation using one hot encoding

evaluate-level1

Explain and compute basic performance metrics for different data science tasks

- [ ] apply at least one metric
- [ ] interpret model performance in context

evaluate-level2

Apply and interpret basic model evaluation metrics to a held out test set

- [ ] apply at least three performance metrics to models
- [ ] apply metrics to subsets of data
- [ ] apply disparity metrics
- [ ] interpret at least three metrics

evaluate-level3

Skip to main content

- [ ] explain cross validation
- [ ] explain importance of held out test and validation data
- [ ] describe why cross vaidation is important
- [ ] idenitfy appropriate metrics for different types of modeling tasks
- [ ] use multiple metrics together to create a more complete description of a model's performance

classification-level1

identify and describe what classification is, apply pre-fit classification models

- [ ] describe what classification is
- [ ] describe what a dataset must look like for classifcation
- [ ] identify appliations of classifcation in the real world
- [ ] describe set up for a classification problem (tes,train)

classification-level2

fit, apply, and interpret preselected classification model to a dataset

- [ ] split data for training and testing
- [ ] fit a classification model
- [ ] apply a classification model to obtain predictions
- [ ] interpret the predictions of a classification model
- [ ] examine parameters of at least one fit classifier to explain how the prediction is made
- [ ] differentiate between model fitting and generating predictions
- [ ] evaluate how model parameters impact model performance

classification-level3

fit and apply classification models and select appropriate classification models for different contexts

- [ ] choose appropriate classifiers based on application context
- [ ] explain how at least 3 different classifiers make predictions
- [ ] evaluate how model parameters impact model performance and justify choices when tradeoffs are necessary

regression-level1

identify what data that can be used for regression looks like

- [ ] identify data that is/not appropriate for regression
- [ ] describe univariate linear regression
- [ ] identify appliations of regression in the real world

regression-level2

Skip to main content

fit and interpret linear regression models

- [ ] split data for training and testing
- [ ] fit univariate linear regression models
- [ ] interpret linear regression models
- [ ] fit multivariate linear regression models

regression-level3

fit and explain regrularized or nonlinear regression

- [ ] fit nonlinear or regrularized regression models
- [ ] interpret and explain nonlinear or regrularized regresion models

clustering-level1

describe what clustering is

- [ ] differentiate clustering from classification and regression
- [ ] identify appliations of clustering in the real world

clustering-level2

apply basic clustering

- [ ] fit Kmeans
- [ ] interpret kmeans
- [ ] evaluate clustering models

clustering-level3

apply multiple clustering techniques, and interpret results

- [ ] apply at least two clustering techniques
- [ ] explain the differences between two clustering models

optimize-level1

Identify when model parameters need to be optimized

- [ ] identify when parameters might impact model performance

optimize-level2

Optimize basic model parameters such as model order

- [ ] evaluate potential tradeoffs
- [ ] interpret optimization results in context

optimize-level3

Select optimal parameters based of mutiple quanttiateve criteria and automate parameter tuning

- [ ] optimize models based on multiple metrics
- [ ] describe when one model vs another is most appropriate

compare-level1

Qualitatively compare model classes

- [ ] compare models within the same task on complexity

compare-level2

Compare model classes in specific terms and fit models in terms of traditional model performance metrics

- [ ] compare models in multiple terms
- [ ] interpret cross model comparisons in context

compare-level3

Evaluate tradeoffs between different model comparison types

- [ ] compare models on multiple criteria
- [ ] compare optimized models
- [ ] jointly interpret optimization result and compare models
- [ ] compare models on quanttiateve and qualitative measures

representation-level1

Identify options for representing text and categorical data in many contexts

- [ ] describe the basic goals for changing the representation of data

representation-level2

Apply at least one representation to transform unstructured or inappropriately data for model fitting or summarizing

- [ ] transform text or image data for use with ML

representation-level3

apply transformations in different contexts OR compare and contrast multiple representations a single type of data in terms of model performance

- [ ] transform both text and image data for use in ml
- [ ] evaluate the impact of representation on model performance

workflow-level1

Solve well strucutred fully specified problems with a single tool pipeline

- [ ] pseudocode out the steps to answer basic data science questions

workflow-level2

Solve well-strucutred, open-ended problems, apply common structure to learn new features of standard tools

- [ ] plan and execute answering real questions to an open ended question
- [ ] describe the necessary steps and tools

workflow-level3

Independently scope and solve realistic data science problems OR independently learn releated tools and describe strengths and weakensses of common tools

- [ ] scope and solve realistic data science problems
- [ ] compare different data science tool stacks

Grading

This section of the syllabus describes the principles and mechanics of the grading for the course. This course will be graded on a basis of a set of *skills* (described in detail the next section of the syllabus). This is in contrast to more common grading on a basis of points earned through assignments.

Principles of Grading

Learning happens through practice and feedback. My goal as a teacher is for you to learn. The grading in this course is based on your learning of the material, rather than your completion of the activities that are assigned.

This course is designed to encourage you to work steadily at learning the material and demonstrating your new knowledge. There are no single points of failure, where you lose points that cannot be recovered. Also, you cannot cram anything one time and then forget it. The material will build and you have to demonstrate that you retained things.

- Earning a C in this class means you have a general understanding of Data Science and could participate in a basic conversation about all of the topics we cover. I expect everyone to reach this level.
- Earning a B means that you could solve simple data science problems on your own and complete parts of more complex problems as instructed by, for example, a supervisor in an internship or entry level job. This is a very accessible goal, it does

- Earning an A means that you could solve moderately complex problems independently and discus the quality of others' data science solutions. This class will be challenging, it requires you to explore topics a little deeper than we cover them in class, but unlike typical grading it does not require all of your assignments to be near perfect.

Grading this way also is more amenable to the fact that there are correct and incorrect ways to do things, but there is not always a single correct answer to a realistic data science problem. Your work will be assessed on whether or not it demonstrates your learning of the targeted skills. You will also receive feedback on how to improve.

How it works

There are 15 skills that you will be graded on in this course. While learning these skills, you will work through a progression of learning. Your grade will be based on earning 45 achievements that are organized into 15 skill groups with 3 levels for each.

These map onto letter grades roughly as follows:

- If you achieve level 1 in all of the skills, you will earn at least a C in the course.
- To earn a B, you must earn all of the level 1 and level 2 achievements.
- To earn an A, you must earn all of the achievements.

You will have at least three opportunities to earn every level 2 achievement. You will have at least two opportunities to earn every level 3 achievement. You will have three *types* of opportunities to demonstrate your current skill level: participation, assignments, and a portfolio.

This is a ha
student wh
below a C.
level 3s an
any work i

Each level of achievement corresponds to a phase in your learning of the skill:

- To earn level 1 achievements, you will need to demonstrate basic awareness of the required concepts and know approximately what to do, but you may need specific instructions of which things to do or to look up examples to modify every step of the way. You can earn level 1 achievements in class, assignments, or portfolio submissions.
- To earn level 2 achievements you will need to demonstrate understanding of the concepts and the ability to apply them with instruction after earning the level 1 achievement for that skill. You can earn level 2 achievements in assignments or portfolio submissions.
- To earn level 3 achievements you will be required to consistently execute each skill and demonstrate deep understanding of the course material, after achieving level 2 in that skill. You can earn level 3 achievements only through your portfolio submissions.

For each skill these are defined in the Achievement Definition Table

Participation

While attending synchronous class sessions, there will be understanding checks and in class exercises. Completing in class exercises and correctly answering questions in class can earn level 1 achievements. In class questions will be administered through the classroom chat platform Prismia.chat; these records will be used to update your skill progression. You can also earn level 1 achievements from adding annotation to a section of the class notes.

Assignments

For your learning to progress and earn level 2 achievements, you must practice with the skills outside of class time.

Skip to main content

Assignments will each evaluate certain skills. After your assignment is reviewed, you will get qualitative feedback on your work, and an assessment of your demonstration of the targeted skills.

Portfolio Checks

To earn level 3 achievements, you will build a portfolio consisting of reflections, challenge problems, and longer analyses over the course of the semester. You will submit your portfolio for review 4 times. The first two will cover the skills taught up until 1 week before the submission deadline.

The third and fourth portfolio checks will cover all of the skills. The fourth will be due during finals. This means that, if you have earned all achievements by the 3rd portfolio check, you do not need to submit the fourth one.

The easiest way to succeed at your portfolio is to extend your assignments

TLDR

You *could* earn a C through in class participation alone, if you make nearly zero mistakes. To earn a B, you must complete assignments and participate in class. To earn an A you must participate, complete assignments, and build a portfolio.

Detailed mechanics

The table below shows the minimum number of skills at each level to earn each letter grade.

letter grade	Level 3	Level 2	Level 1
A	15	15	15
A-	10	15	15
B+	5	15	15
B	0	15	15
B-	0	10	15
C+	0	5	15
C	0	0	15
C-	0	0	10
D+	0	0	5
D	0	0	3

For example, if you achieve level 2 on all of the skills and level 3 on 7 skills, that will be a B+.

If you achieve level 3 on 14 of the skills, but only level 1 on one of the skills, that will be a B-, because the minimum number of level 2 achievements for a B is 15. In this scenario the total number of achievements is 14 at level 3, 14 at level 2 and 15 at level 3, because you have to earn achievements within a skill in sequence.

The letter grade can be computed as follows

Skip to main content

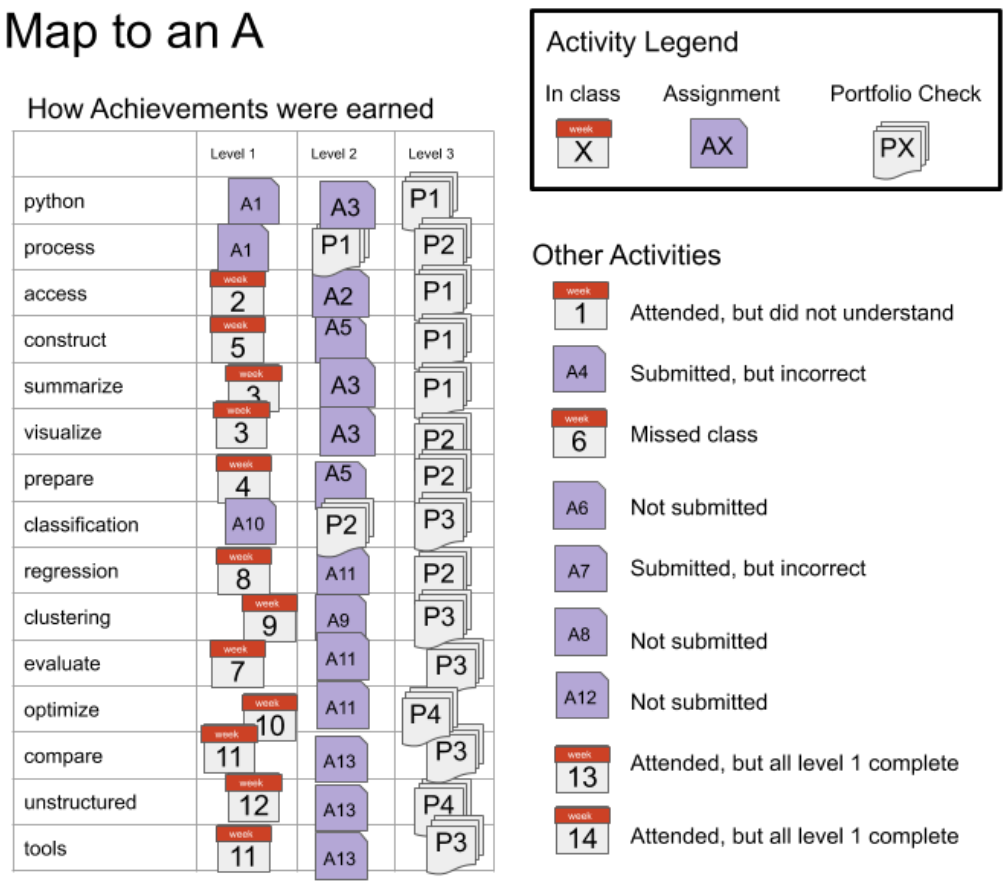
Important

this will be revealed after assignment 1

Grading Examples

If you always attend and get everything correct, you will earn and A and you won't need to submit the 4th portfolio check.

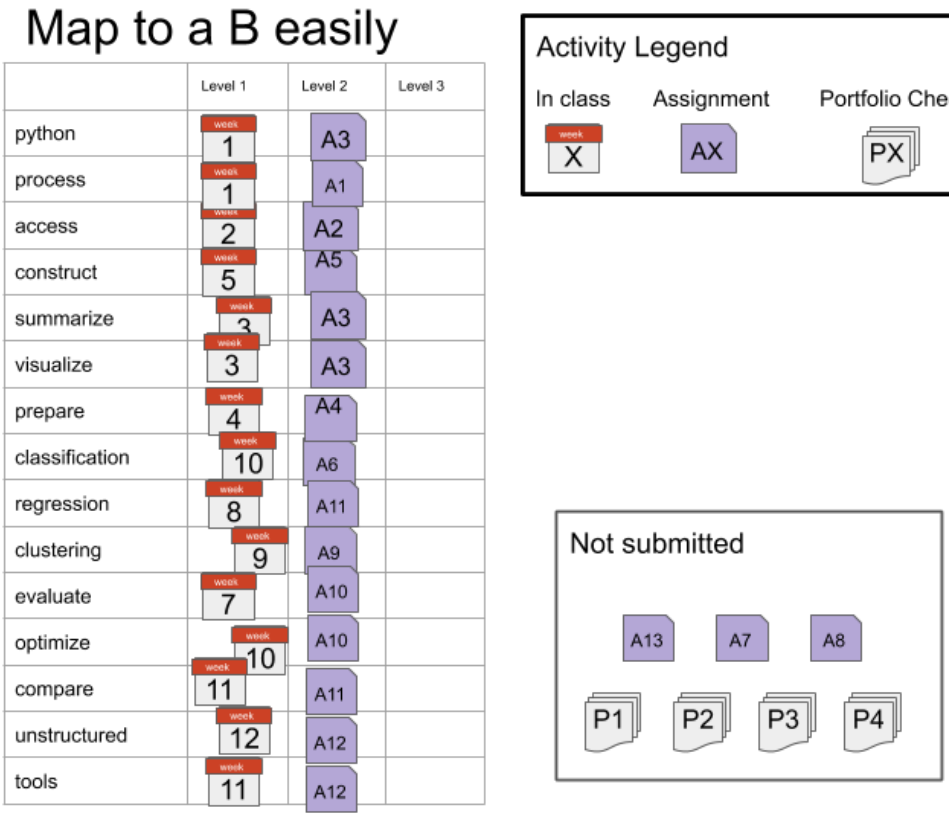
Getting an A Without Perfection



In this example the student made several mistakes, but still earned an A. This is the advantage to this grading scheme. For the `python`, `process`, and `classification` skills, the level 1 achievements were earned on assignments, not in class. For the `process` and `classification` skills, the level 2 achievements were not earned on assignments, only on portfolio checks, but they were earned on the first portfolio of those skills, so the level 3 achievements were earned on the second portfolio check for that skill. This student's fourth portfolio only demonstrated two skills: `optimize` and `unstructured`. It included only 1 analysis, a text analysis with optimizing the parameters of the model. Assignments 4 and 7 were both submitted, but didn't earn any achievements, the student got feedback though, that they were able to apply in later assignments to earn the achievements. The student missed class week 6 and chose to not submit assignment 6 and use week 7 to catch up. The student had too much work in another class and chose to skip assignment 8. The student tried assignment 12, but didn't finish it on time, so it was not graded, but the student visited office hours to understand and be sure to earn the level 2 `unstructured` achievement on assignment 13.

Getting a B with minimal work

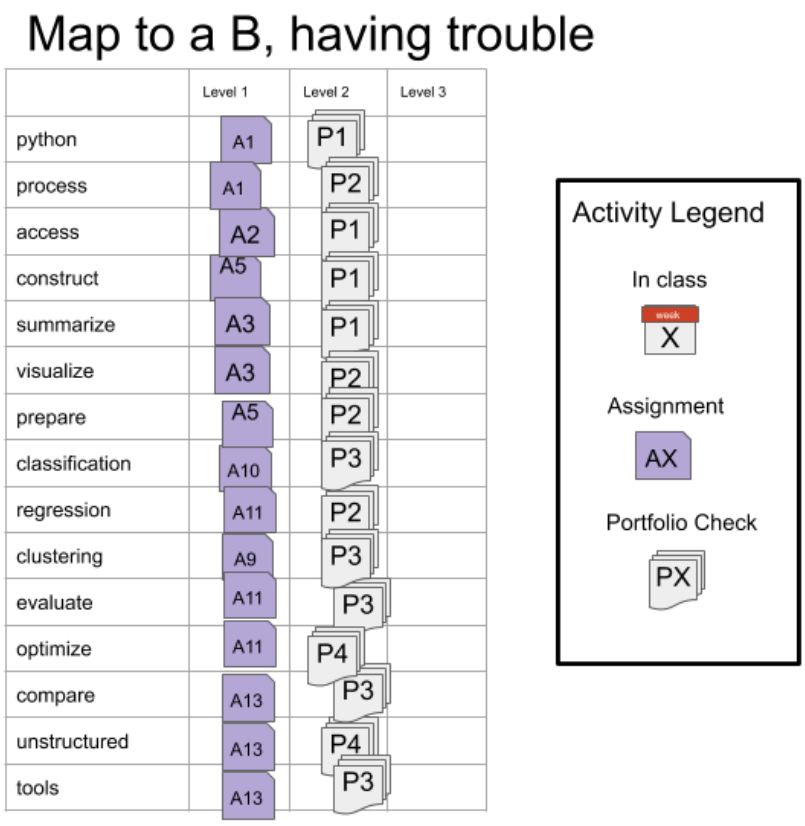
Skip to main content



In this example, the student earned all level 1 achievements in class and all level 2 on assignments. This student was content with getting a B and chose to not submit a portfolio.

Getting a B while having trouble

Skip to main content



In this example, the student struggled to understand in class and on assignments. Assignments were submitted that showed some understanding, but all had some serious mistakes, so only level 1 achievements were earned from assignments. The student wanted to get a B and worked hard to get the level 2 achievements on the portfolio checks.

Grading Policies

Attendance

Attendance and active participation is expeted. You earn level 1 achievements in class and all class sessions are active learning.

If you miss class, you can make it up by reading the posted notes and the prismia transcript. Best practice is to download them as a notebook and run them to make sure you understand each step. If you miss both class sessions in a week, the level one achievements can be made up through annotation or in your assignment.

Absences do not require notification.

Assignment Deadlines and Late Work

Late assignments will not be graded. Extensions will not be granted for assignments. Every skill will be assessed through more than one assignment, so missing assignments occasionally will not necessarily impact your grade. If you do not submit **any** assignments that cover a given skill, you may earn the level 2 achievement in that skill through a portfolio check, but you will have fewer chances to earn level 3 in that skill.

Skip to main content

If you submit work that is **not complete**, it will be assessed and receive feedback. Submitting pseudocode or code with errors and comments about what you have tried could even be enought earn a level 1 achievement. Assignments cover multiple skills, so partially completing the assignment may earn level 2 for one, but not all. Submitting *something* even if it is not perfect is important to keeping conversation open and getting feedback and help continuously.

> **Important**
>
> If you have a serious issue during the semester, that prevents you from submitting an assignment, email Dr. Brown to make a plan. Extensions will still not be granted because they do not help you in the long run, instead an alternate plan of how to earn the target grade.

Portfolio Deadlines and Extensions

Building your Data Science Portfolio should be an ongoing process, where you commit work to your portfolio frequently. If something comes up and you cannot finish all that you would like assessed by the deadline, open an `Extension Request` issue on your repository at least **24 hours** before the deadline.

In this issue, include:

1. A proposed new deadline
2. What additional work you plan to add
3. Why the extension is important to your learning
4. Why the extension will not hinder your ability to complete the next assignments and portfolio check on time.
5. (if less than 24 hours before the deadline) why you need an emergency request

> **Important**
>
> Your request should not include a reason why you are asking, unless you are asking for an emergency extension. Emergency requests can be submitted at any time, even after the deadline.

This request should be no more than 7 sentences.

Portfolio due dates will be announced well in advance and prompts for it will be released weekly. You should spend some time working on it each week, applying what you've learned so far, from the feedback on previous assignments.

Academic Dishonesty

All work must represent your own understanding of both the data science practices and the related programming concepts. Submitting code or prose that was generated by a generative model or another person is not allowed.

If you are found to have submitted work that does not constitute your own work, the following penalties apply:

- in a portfolio, all achievements attempted in the dishonest component are permanently ineligible.
- in an assignment the level three achievements for the skills of focus in the assignment are ineligible, and the relevent level two for those skills requires meeting the standard for the level 3.

Skip to main content

If you violate acadmic honesty policy in portfolio 1 while attempting level 3 at Python, access, prepare, summarize and visualize and process level 2, then your maximum grade becomes a B+, because level 3 in all five of those skills becomes inelgible.

Regrading

1. Add comments:
    - For general questions, post on the conversation tab of your Feedback PR with your request.
    - For specific questions, reply to a specifc comment.
2. Re-request a review from Dr. Brown on your Feedback Pull request.

If you think we missed *where* you did something, add a comment on that line to help us find it (on the code tab of the PR, click the plus (+) next to the line) and then post on the conversation tab with an overview of what you're requesting and tag @brownsarahm

Course Style Guide

Following a style guide is a common requirement in companies to make it so that code written by different people stays easy to read for everyone. Consistent style also makes it easier to onboard new developers join a project and contribute faster.

The following style guide serves as practice for you following a style guide, makes your work easier to read for grading purposes, and holds you accountable to learning deeply and demonstrating that you have learned well.

Hard Requirements

> **Warning**
>
> All work must adhere to these requirements or it may receive no feedback or credit. Minor misses may receive warnings, but if submitted work does not appear to represent a good faith effort at adhering to this style guide, the only comment will be, "Follow the style guide on the next assignment"

1. All code must be submitted in a notebook file (.ipynb or myst)
2. Code must run or have explicit questions and comments about what was done about the errors
3. Python comments (`# comment text`) inside code cells should **only** be used to explain complex code that is not explained in the course notes. Using such code requires a citation for the source.
4. Each code cell should represent at most one conceptually complete step in terms of the analysis.
5. Every code cell must be motivated by text in markdown before it
6. Every code cell's output must be interpretted
7. the `print` function can only be used when it improves the readability over using jupyter's display, must be justified
8. No deprecated or dangerous code constructs without justification
9. All assignment questions must be answered in markdown cells
10. Notebook files may not have extraneous metadata in them

Additional Style

Skip to main content

Important

Mistakes on these will get detailed feedback once and a “see previous feedback” a second time before the whole assignment receives no feedback.

1. Code should ahdere to PEP8
2. Markdown syntax should be used to enhance the readability of the text (eg not all headings, bullets where they make sense)
3. Best practices that are highlighted in class should be followed (this list will expand over the semester)

Support

Warning

URI changed some links and this page is not yet up to date

Academic Enhancement Center

Academic Enhancement Center (for undergraduate courses): Located in Roosevelt Hall, the AEC offers free face-to-face and web-based services to undergraduate students seeking academic support. Peer tutoring is available for STEM-related courses by appointment online and in-person. The Writing Center offers peer tutoring focused on supporting undergraduate writers at any stage of a writing assignment. The UCS160 course and academic skills consultations offer students strategies and activities aimed at improving their studying and test-taking skills. Complete details about each of these programs, up-to-date schedules, contact information and self-service study resources are all available on the AEC website.

- **STEM Tutoring** helps students navigate 100 and 200 level math, chemistry, physics, biology, and other select STEM courses. The STEM Tutoring program offers free online and limited in-person peer-tutoring this fall. Undergraduates in introductory STEM courses have a variety of small group times to choose from and can select occasional or weekly appointments. The TutorTrac application is available through URI Microsoft 365 single sign-on and by visiting aec.uri.edu. More detailed information and instructions can be found on the AEC tutoring page.
- **Academic Skills Development** resources helps students plan work, manage time, and study more effectively. In Fall 2020, all Academic Skills and Strategies programming are offered both online and in-person. UCS160: Success in Higher Education is a one-credit course on developing a more effective approach to studying. Academic Consultations are 30-minute, 1 to 1 appointments that students can schedule on Starfish with Dr. David Hayes to address individual academic issues. Study Your Way to Success is a self-guided web portal connecting students to tips and strategies on studying and time management related topics. For more information on these programs, visit the Academic Skills Page or contact Dr. Hayes directly at davidhayes@uri.edu.
- The **Undergraduate Writing Center** provides free writing support to students in any class, at any stage of the writing process: from understanding an assignment and brainstorming ideas, to developing, organizing, and revising a draft. Fall 2020 services are offered through two online options: 1) real-time synchronous appointments with a peer consultant (25- and 50-minute slots, available Sunday - Friday), and 2) written asynchronous consultations with a 24-hour turn-around response time (available Monday - Friday). Synchronous appointments are video-based, with audio, chat, document-sharing, and live captioning capabilities, to meet a range of accessibility needs. View the synchronous and asynchronous schedules and book online, visit uri.mywconline.com.

Warning

URI changed some links and this page is not yet up to date

Anti-Bias Statement:

We respect the rights and dignity of each individual and group. We reject prejudice and intolerance, and we work to understand differences. We believe that equity and inclusion are critical components for campus community members to thrive. If you are a target or a witness of a bias incident, you are encouraged to submit a report to the URI Bias Response Team at www.uri.edu/brt. There you will also find people and resources to help.

Mental Health and Wellness

We understand that college comes with challenges and stress associated with your courses, job/family responsibilities and personal life. URI offers students a range of services to support your mental health and wellbeing, including the URI Counseling Center, MySSP (Student Support Program) App, the Wellness Resource Center, and Well-being Coaching.

Disability Services for Students Statement:

Your access in this course is important. Please send me your Disability Services for Students (DSS) accommodation letter early in the semester so that we have adequate time to discuss and arrange your approved academic accommodations. If you have not yet established services through DSS, please contact them to engage in a confidential conversation about the process for requesting reasonable accommodations in the classroom. DSS can be reached by calling: 401-874-2098, visiting: web.uri.edu/disability, or emailing: dss@etal.uri.edu. We are available to meet with students enrolled in Kingston as well as Providence courses.

Academic Honesty

Students are expected to be honest in all academic work. A student's name on any written work, quiz or exam shall be regarded as assurance that the work is the result of the student's own independent thought and study. Work should be stated in the student's own words, properly attributed to its source. Students have an obligation to know how to quote, paraphrase, summarize, cite and reference the work of others with integrity. The following are examples of academic dishonesty.

- Using material, directly or paraphrasing, from published sources (print or electronic) without appropriate citation
- Claiming disproportionate credit for work not done independently
- Unauthorized possession or access to exams
- Unauthorized communication during exams
- Unauthorized use of another's work or preparing work for another student
- Taking an exam for another student
- Altering or attempting to alter grades
- The use of notes or electronic devices to gain an unauthorized advantage during exams
- Fabricating or falsifying facts, data or references directly or indirectly through the use of generative AI
- Facilitating or aiding another's academic dishonesty
- Submitting the same paper for more than one course without prior approval from the instructors

Skip to main content

Communications & Office Hours

Announcements

Announcements will be made via GitHub Release. You can view them online in the releases page or you can get notifications by watching the repository, choosing “Releases” under custom see GitHub docs for instructions with screenshots. You can choose GitHub only or e-mail notificaiton from the notification settings page

Help Hours

Day	Time	Location	Host
Monday	12pm-2pm	Zoom	Mark
Monday	4-5pm	Zoom	Dr. Brown
Friday	4-5pm	134 Tyler	Dr. Brown

We have several different ways to communicate in this course. This section summarizes them

To reach out, By usage

usage	platform	area	note
in class	prismia	chat	outside of class time this is not monitored closely
any time	prismia	download transcript	use after class to get preliminary notes eg if you miss a class
private questions to your assignment	github	issue on assignment repo	eg bugs in your code"
for general questions that can help others	github	issue on course website	eg what the instructions of an assignment mean or questions about the syllabus
to share resources or ask general questions in a semi-private forum	github	discussion on community repo	include links in your portfolio
matters that don't fit into another category	e-mail	to brownsarahm@uri.edu	remember to include `[CSC310]` or `[DSP310]` (note `verbatim` no space)

Note

e-mail is last because it’s not collaborative; other platforms allow us (Proessor + TA) to collaborate on who responds to things more easily.

Tips

For assignment help

- **send in advance, leave time for a response** I check e-mail/github a small number of times per day, during work hours, almost exclusively. You might see me post to this site, post to BrightSpace, or comment on your assignments outside of my normal working hours, but I will not reliably see emails that arrive during those hours. This means that it is important to start assignments early.

Using issues

- use issues for content directly related to assignments. If you push your code to the repository and then open an issue, I can see your code and your question at the same time and download it to run it if I need to debug it
- use issues for questions about this syllabus or class notes. At the top right there's a GitHub logo that allows you to open a issue (for a question) or suggest an edit (eg if you think there's a typo or you find an additional helpful resource related to something)

For E-email

- use e-mail for general inquiries or notifications
- Please include `[CSC310]` or `[DSP310]` in the subject line of your email along with the topic of your message. This is important, because your messages are important, but I also get a lot of e-mail. Consider these a cheat code to my inbox: I have setup a filter that will flag your e-mail if you use one of those in the subject to ensure that I see it.

Not

Whet
not m

1. Welcome & What is Data Science

1.1. Prismia Chat

We will use these to monitor your participation in class and to gather information. Features:

- instructor only
- reply to you directly
- share responses for all

1.2. How this class will work

Participatory Live Coding

What is a topic you want to use data to learn about?

Debugging is both technical and a soft skill

1.3. Programming for Data Science vs other Programming

The audience is different, so the form is different.

In Data Science our product is more often a report than a program.

Skip to main content

Sometimes there will be points in the notes that were not made in class due to time or in response questions that came at the end of class.

Also, in data science we are *using code* to interact with data, instead of having a plan in advance

So programming for data science is more like *writing* it has a narrative flow and is made to be seen more than some other programming thaat you may have done.

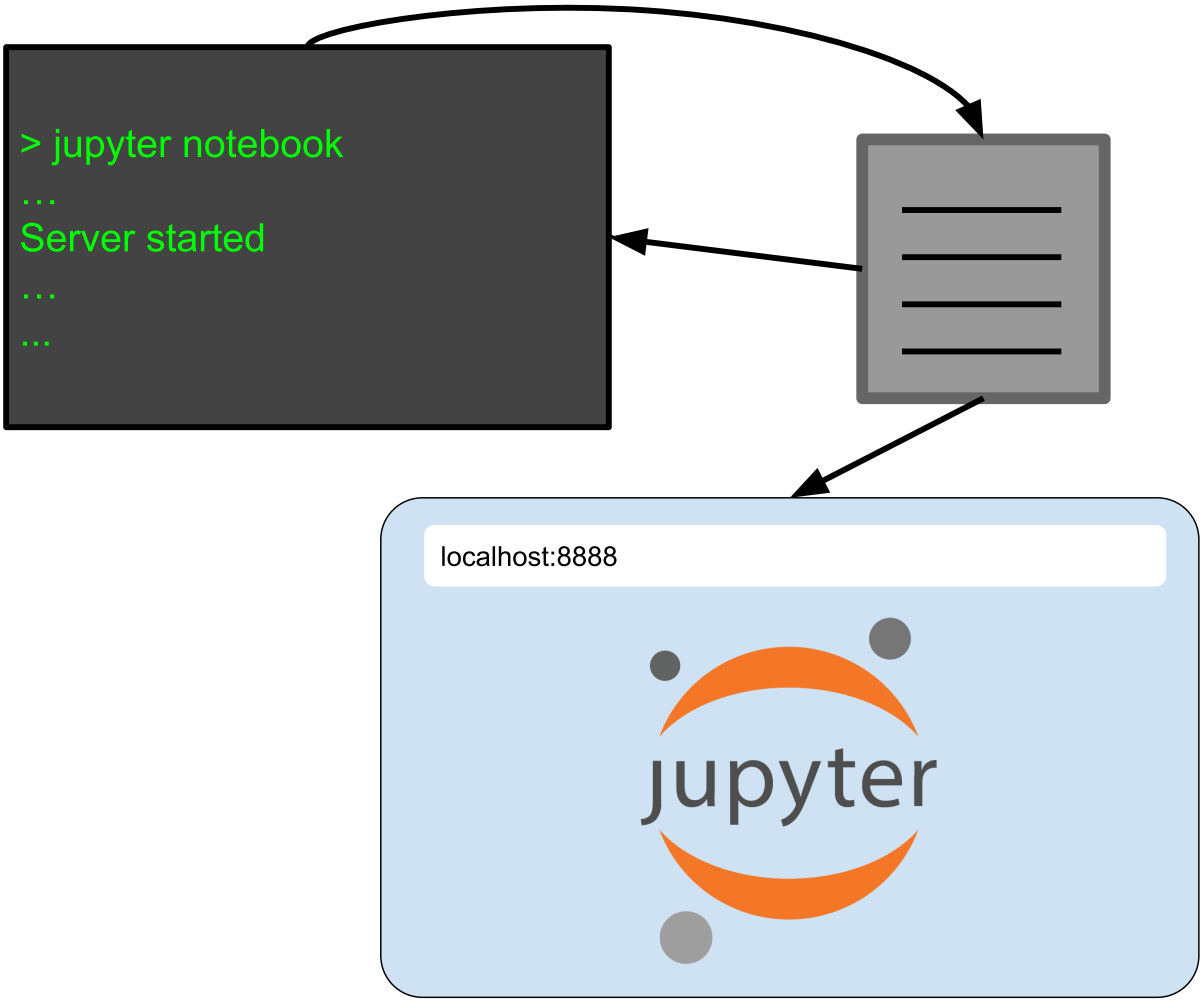
1.4. Jupyter Lab and Jupyter notebooks

Launch a `jupyter lab` server:

- on Windows, use anaconda terminal
- on Mac/Linux, use terminal
- `cd path/to/where/you/save/notes`
- enter `jupyter lab`

1.4.1. What just happened?

- launched a local web server
- opened a new browser tab pointed to it



1.4.2. A jupyter notebook tour

A Jupyter notebook has two modes. When you first open, it is in command mode. It says the mode in the bottom right of the screen. Each bos is a cell, the highlighted cell is gray when in command mode.

When you press a key in command mode it works like a shortcut. For example `p` shows the command search menu.

If you press `enter` (or `return`) or click on the highlighted cell, which is the boxes we can type in, it changes to edit mode.

There are two type of cells that we will used: code and markdown. You can change that in command mode with `y` for code and `m` for markdown or on the cell type menu at the top of the notebook.

This is a markdown cell

- we can make
- itemized lists of
- bullet points

1. and we can make nubmered
2. lists, and not have to worry

Skip to main content

4. if we add a step in the middle later

```
# this is a comment in a code cell
3+9
```

```
12
```

the output here is the value returned by the python interpretter for the last line of the cell

We can set variables

```
name = 'sarah'
```

The notebook displays nothing when we do an assignment, bcause it returns nothing

we can put a variable there tosee it

```
name
```

```
'sarah'
```

```
name
course = 'csc310'
```

it only does that for the last line, so this one displays nothing

> **Important**
>
> In class, we ran these cells out of order and noticed how the value does not update unless we run the new version

```
name*3
```

```
'sarahsarahsarah'
```

Common command mode actions:

- m: switch cell to markdown
- y: switch cell to code
- a: add a cell above
- b: add a cell below
- c: copy cell
- v: paste the cell
- 0 + 0: restart kernel
- h: command menu

use enter/return to get to edit mode

1.5. Getting Help in Jupyter

Getting help is important in programming

When your cursor is inside the `()` of a function if you hold the shift key and press tab it will open a popup with information. If you press tab twice, it gets bigger and three times will make a popup window.

Python has a `print` function and we can use the help in jupyter to learn about how to use it in different ways.

```
print(name,course)
```

```
sarah csc310
```

The first line says that it can take multiple values, because it says `args*, sep`. The `*` means multiple.

It also has a keyword argument (must be used like `argument=value` and has a default) described as `sep=' '`. This means that by default it adds a space as above.

The help also tells us about other parameters, like the `sep` one

```
print(name,course,sep="_")
```

```
sarah_csc310
```

We can print the docstring out, as a whole instead of using the shfit + tab to view it.

```
help(print)
```

```
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.
```

This looks similar to the one above

```
print(name + '_'+ course)
```

```
sarah_csc310
```

Skip to main content

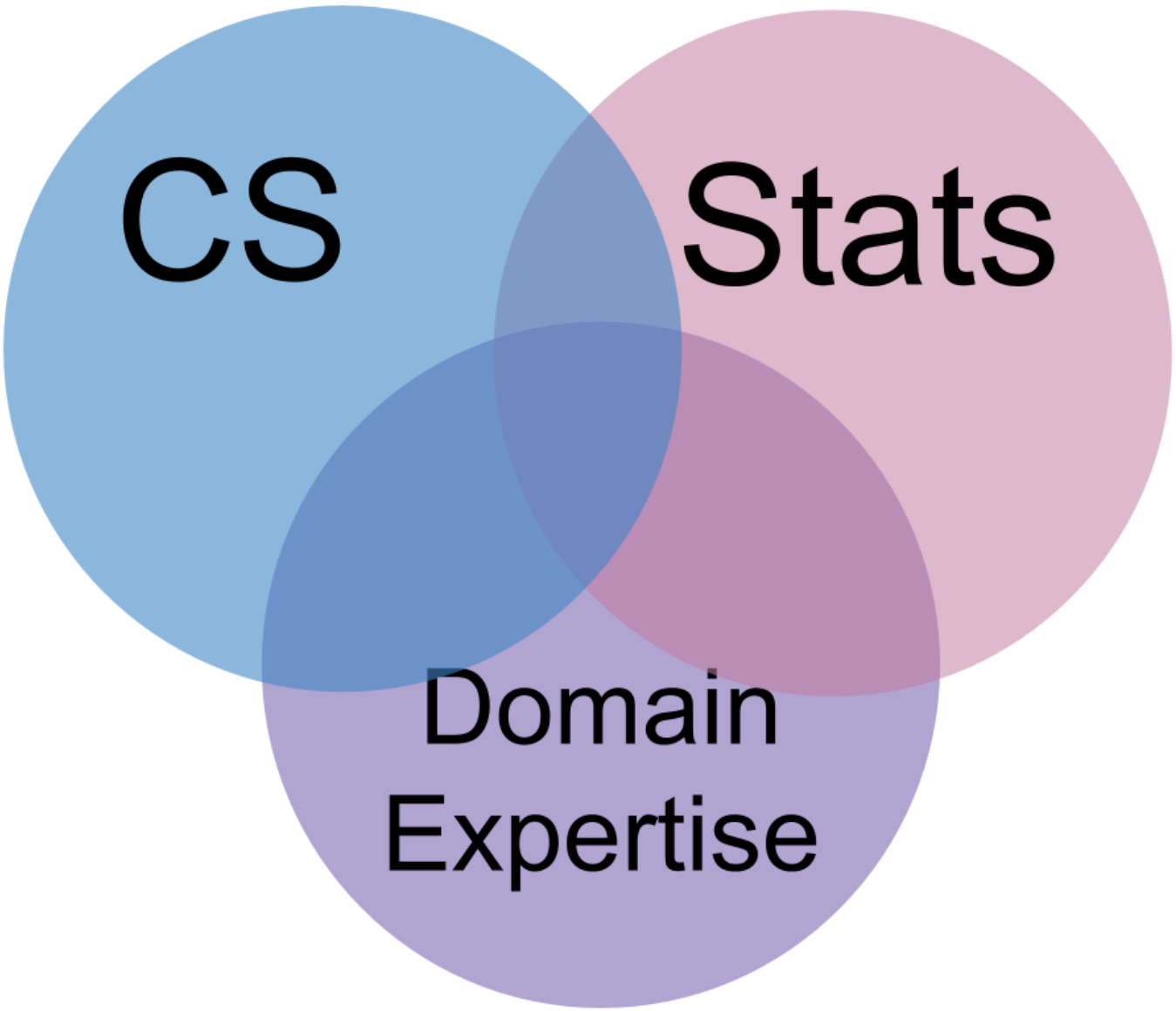
```
print(name,course,'hello',"bye", sep='_')
```

```
sarah_csc310_hello_bye
```

Basic programming is a prereq and we will go faster soon, but the goal of this review was to understand notebooks, getting help, and reading docstrings

1.6. What is Data Science?

Data Science is the combination of



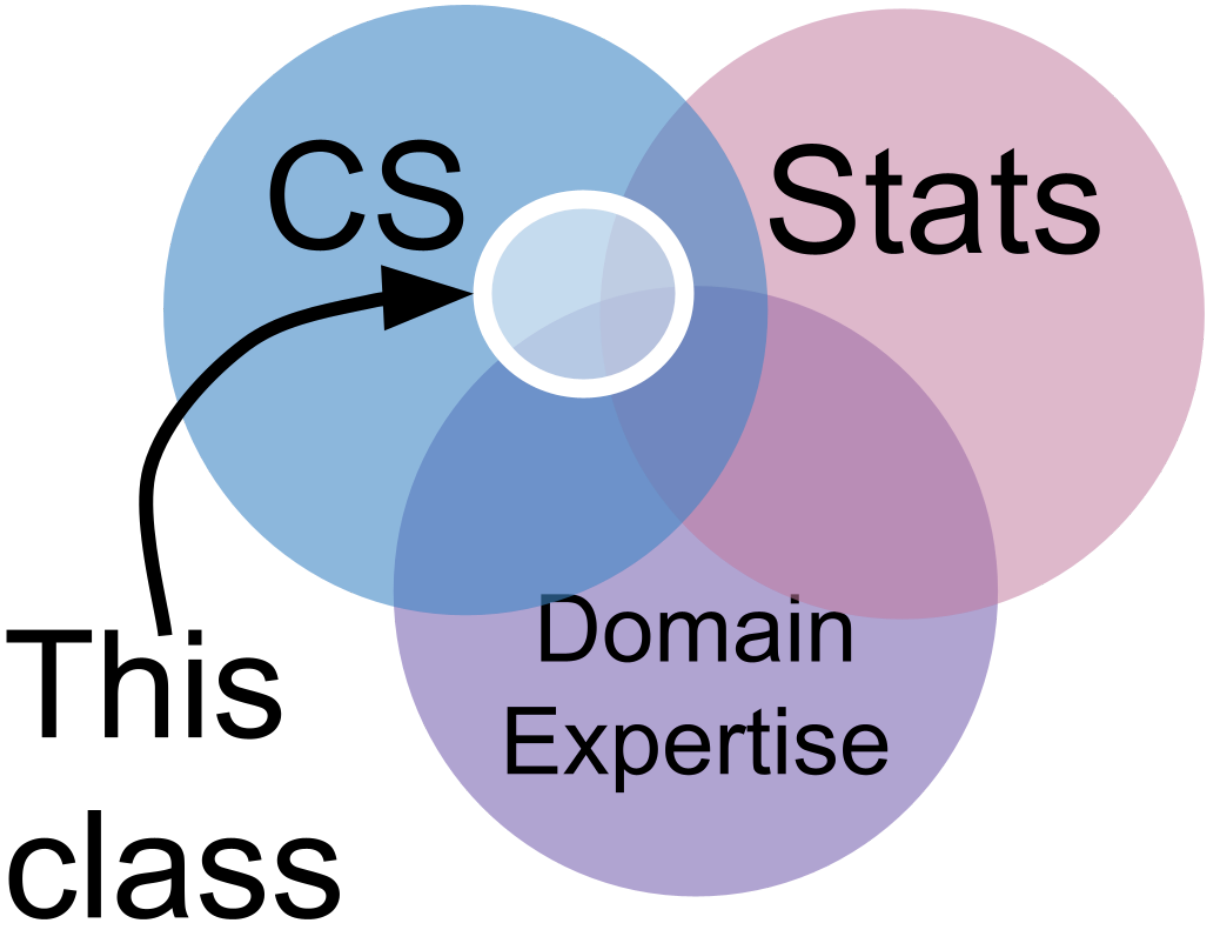
statistics is the type of math we use to make sense of data. Formally, a statistic is just a function of data.

computer science is so that we can manipulate visualize and automate the inferences we make.

domain expertise helps us have the intuition to know if what we did and what we found. A statistic must be interpreted in context of the

it can help us tell whether our code actually worked right or not.

1.6.1. In this class,

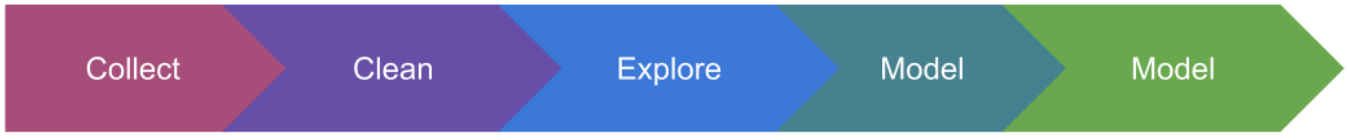


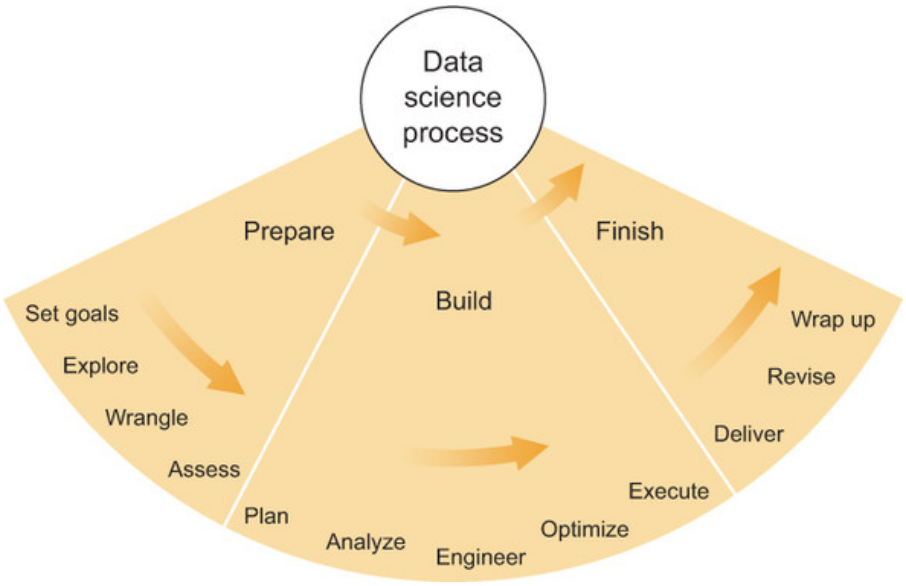
We'll focus on the programming as our main means of studying data science, but we will use bits of the other parts. In particular, you're encouraged to choose datasets that you have domain expertise about, or that you want to learn about.

But there are many definitions. We'll use this one, but you may come across others.

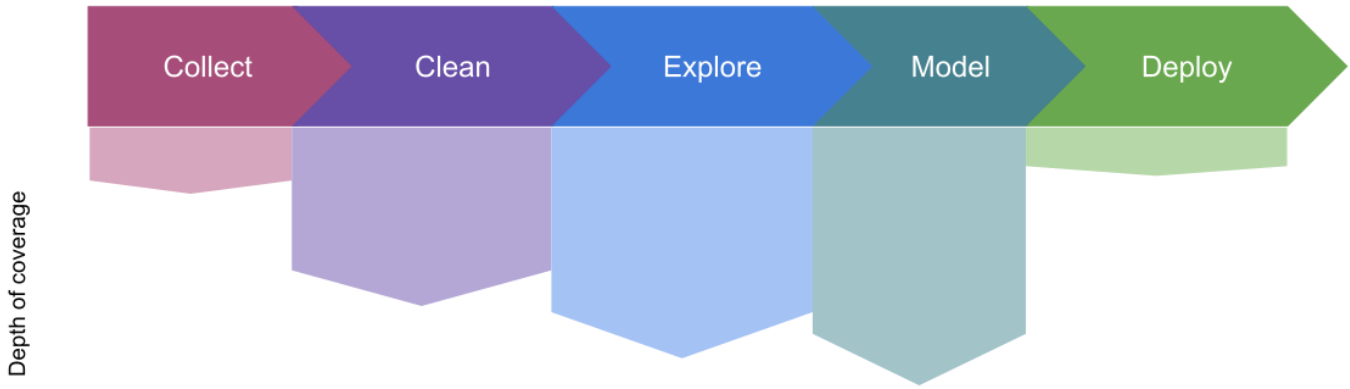
1.6.2. How does data science happen?

The most common way to think about what doing data science means is to think of this pipeline. It is in the perspective of the data, these are all of the things that happen to the data.



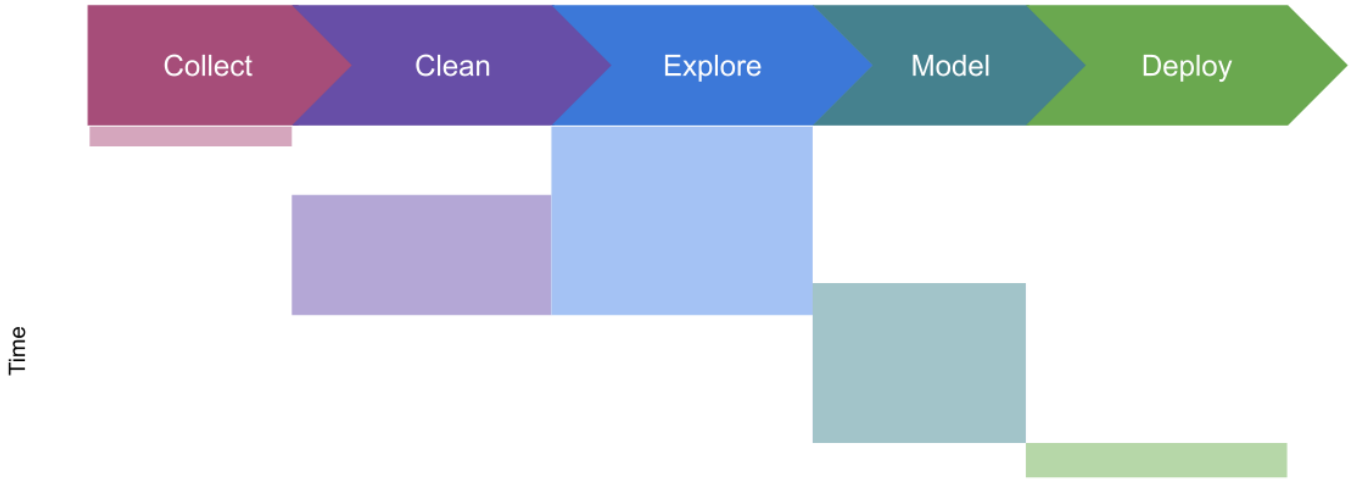


1.6.3. how we'll cover Data Science, in depth



- *collect*: Discuss only a little; Minimal programming involved
- *clean*: Cover the main programming techniques; Some requires domain knowledge beyond scope of course
- *explore*: Cover the main programming techniques; Some requires domain knowledge beyond scope of course
- *model*:Cover the main programming, basic idea of models; How to use models, not how learning algorithms work
- *deploy*: A little bit at the end, but a lot of preparation for decision making around deployment

1.6.4. how we'll cover it in, time



We'll cover exploratory data analysis before cleaning because those tools will help us check how we've cleaned the data.

1.7. Python Review

Official source on python:

- pep8 official style
- documentation note that you can change which version you are using

We will go quickly through these focusing on pythonic style, because the prerequisite is a programming course.

1.8. Functions

```
def greeting(name):
    '''
    say hi to a person

    Parameters
    +++++++++-
    name : string
        the name of the person to greet
    '''
    return 'hi ' + name
```

A few things to note:

- the `def` keywords starts a function
- then the name of the function
- parameters in `()` then `:`
- the body is indented
- the first thing in the body should be a docstring, denoted in `'''` which is a multiline comment
- returning is more reliable than printing in a function

In python, PEP 257 says how to write a docstring, but it is very broad.

Skip to main content

In Data Science, numpydoc style docstrings are popular.

- Pandas follows numpydoc
- [Numpy uses it]
- Scipy follows numpydoc

Once the cell with the function definition is run, we can use the function

```
greeting(name)
```

```
'hi sarah'
```

```
print(greeting('surbhi'))
```

```
hi surbhi
```

```
assert greeting('sarah') == 'hi sarah'
```

With a return this works to check that it does the right thing.

when assert is true, it returns nothing, it throws an error on failure

1.9. Conditionals

```
def greeting2(name,formal=False):
    '''
    say hi to a person

    Parameters
    +++++++++-
    name : string
        the name of the person to greet
    formal: bool
        if the greeting should formal (hello) or not (hi)
    '''
    if formal:
        message = 'hello  ' + name
    else:
        message = 'hi ' + name
    return message
```

key points in this function:

- an `if` also has the conditional part indented
- for a `bool` variable we can just use the variable
- we can set a default value

because of the default value we do not have to pass the second variable:

```
'hi sarah'
```

```
greeting2(name,True)
```

```
'hello  sarah'
```

1.10. Hints

Reading chapter 1 of think like a data scientist will help you with the data science definition part of the assignment.

Think like a data scientist is written for practitioners; not as a text book for a class. It does not have a lot of prerequisite background, but the sections of it that I assign will help you build a better mental picture of what doing Data Science about.

Only the first assignment will be due this fast, it's a short review and setup assignment. It's due quickly so that we know that you have everything set up and the prerequisite material before we start new material next week.

2. Iterables and Pandas Data Frames

2.1. House Keeping

2.1.1. Grading is not done,

you will get a notificaiton when yours is

2.1.2. Closing Jupyter server.

In the terminal use Ctrl+C (actually control, not command on mac).

It will ask you a question and give options, read and follow

or

do ctrl+C a second time.

A jupyter server typically runs at `localhost:8888`, but if you have multiple servers running the count increases.

Once I saw a student in office hours working on `localhost:8894` asking why their code kept crashing.

> **Important**
>
> Remember to close your jupyter server

2.2. Grading solution

Skip to main content

```
def compute_grade(num_level1,num_level2,num_level3):
    '''
    Computes a grade for CSC/DSP310 from numbers of achievements at each level

    Parameters:
    -----------

    num_level1 : int
      number of level 1 achievements earned
    num_level2 : int
      number of level 2 achievements earned
    num_level3 : int
      number of level 3 achievements earned

    Returns:
    --------
    letter_grade : string
      letter grade with modifier (+/-)
    '''
    if num_level1 == 15:
        if num_level2 == 15:
            if num_level3 == 15:
                grade = 'A'
            elif num_level3 >= 10:
                grade = 'A-'
            elif num_level3 >=5:
                grade = 'B+'
            else:
                grade = 'B'
        elif num_level2 >=10:
            grade = 'B-'
        elif num_level2 >=5:
            grade = 'C+'
        else:
            grade = 'C'
    elif num_level1 >= 10:
        grade = 'C-'
    elif num_level1 >= 5:
        grade = 'D+'
    elif num_level1 >=3:
        grade = 'D'
    else:
        grade = 'F'


    return grade
```

When we run the cell above that adds the function to memory.

Now that it is run, jupyter can show us `compute_grade` as an option when we tab complete after typing the first few letters.

When we restarted the kernel, we saw that before running the cell above, the tab complete did not work.

> **Important**
>
> this is important to understand what works when and why so that you know what to expect and can get unstuck

```
compute_grade(15,15,14)
```

```
'A-'
```

Skip to main content

```
assert compute_grade(15,15,15) =='A'
```

`asssert` succeeds quietly

```
assert compute_grade(15,15,15) =='B'
```

```
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
Cell In[4], line 1
----> 1 assert compute_grade(15,15,15) =='B'

AssertionError:
```

but fails with a specific error

The docstring is important, because it is the help.

```
help(compute_grade)
```

```
Help on function compute_grade in module __main__:

compute_grade(num_level1, num_level2, num_level3)
    Computes a grade for CSC/DSP310 from numbers of achievements at each level

    Parameters:
    -----------

    num_level1 : int
      number of level 1 achievements earned
    num_level2 : int
      number of level 2 achievements earned
    num_level3 : int
      number of level 3 achievements earned

    Returns:
    --------
    letter_grade : string
      letter grade with modifier (+/-)
```

2.3. Everything is Data

Data we will see:

- tabular data
- websites as data
- activity logs on websites
- images
- text

2.4. Why inspection in code?

Skip to main content

Some IDEs give you GUI based tools to inspect objects. We are going to do it programmatically inline with our analyses for two reasons.

- (minor, logistical) it helps make for good notes
- (most importantly) it helps build habits of data science

In data science, our code will be aiming to tell a story.

If you're curious about something, try it out, see what happens. We're going to use a lot of code inspection tools during class. These are helpful both for understanding what's going on, but the advantage to knowing how to get this information programmatically even though a different IDE would give you inspection tools is that it helps you treat your code as data.

2.5. everything is an object

let's examine the `type` of some variables:

```
a = 4
b ='monday'
c = 5.3
d =print
```

```
type(a)
```

```
int
```

ints are a base python type, like they appear in other languages

strings are iterable type, meaning that theycan be indexed into, or their elements iterated over. For a more technical definition, see the official python glossary entry

```
type(b)
```

```
str
```

we can select one element

```
b[0]
```

```
'm'
```

or multiple, this is called slicing.

```
b[0:3]
```

```
'mon'
```

negative numbers count from the right.

```
b[-1]
```

```
'y'
```

decimals defualt to float

```
type(c)
```

```
float
```

a variable can hold a whole function.

```
type(d)
```

```
builtin_function_or_method
```

functions are also objects like any other type in python

we can use the variable just like the function itself

```
d('hello')
```

```
hello
```

```
print('hello')
```

```
hello
```

2.6. Tabular Data

Structured data is easier to work with than other data.

We're going to focus on tabular data for now. At the end of the course, we'll examine images, which are structured, but more complex and text, which is much less structured.

2.7. Getting familiar with the datset

We're going to use a dataset about coffee quality today.

How was this dataset collected?

- reviews added to DB
- then scraped

Where did it come from?

- coffee Quality Institute's trained reviewers.

what format is it provided in?

- csv (Comma Separated Values)

what other information is in this repository?

- the code to scrape and clean the data
- the data before cleaning

It's important to always know where data came from and how it was collected.

This helps you know what is is useful for and what its limitations are.

> **Further Reading**
>
> An important research article on documenting datasets for machine learning is called Datasheets for Datasets these researchers also did a follow up study to better understand how practitioner use datasheets and decide how to use data.
>
> If topics like this are interesting to you, let me know! my research is related to this and I have a lot of students who complete 310 do research in my lab.

2.8. Loading data

Get raw url for the dataset click on the raw button on the csv page, then copy the url.

```
coffee_data_url = 'https://raw.githubusercontent.com/jldbc/coffee-quality-database/master/data/robusta_data_
```

We will use data with a library called pandas. By convention, we import it like:

```
import pandas as pd
```

- the `import` keyword is used for loading packages
- `pandas` is the name of the package that is installed
- `as` keyword allows us to assign an alias (nickname)
- `pd` is the typical alias for pandas

we will load the data with `pd.read_csv()`

```
pd.read_csv(coffee_data_url)
```

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number
0	1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ankole coffee producers	0
1	2	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	25	sethuraman estate	14/1148/2017/21
2	3	Robusta	andrew hetzel	India	sethuraman estate	NaN	NaN	0000
3	4	Robusta	ugacof	Uganda	ugacof project area	NaN	ugacof	0
4	5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katuka development trust	0
5	6	Robusta	andrew hetzel	India	NaN	NaN	(self)	NaN
6	7	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN
7	8	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	7	sethuraman estate	14/1148/2017/18
8	9	Robusta	nishant gurjer	India	sethuraman estate	RKR	sethuraman estate	14/1148/2016/17
9	10	Robusta	ugacof	Uganda	ishaka	NaN	nsubuga umar	0
10	11	Robusta	ugacof	Uganda	ugacof project area	NaN	ugacof	0
11	12	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	RC AB	sethuraman estate	14/1148/2016/12
12	13	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN
13	14	Robusta	kasozi coffee farmers association	Uganda	kasozi coffee farmers	NaN	NaN	0
14	15	Robusta	ankole coffee producers coop	Uganda	kyangundu coop society	NaN	ankole coffee producers coop union ltd	0
15	16	Robusta	andrew hetzel	India	sethuraman estate	NaN	NaN	0000
16	17	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuraman estates	NaN

Skip to main content

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number
17	18	Robusta	kawacom uganda ltd	Uganda	bushenyi	NaN	kawacom	0
18	19	Robusta	nitubaasa ltd	Uganda	kigezi coffee farmers association	NaN	nitubaasa	0
19	20	Robusta	mannya coffee project	Uganda	mannya coffee project	NaN	mannya coffee project	0
20	21	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN
21	22	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuraman estates	NaN
22	23	Robusta	andrew hetzel	United States	sethuraman estates	NaN	sethuraman estates	NaN
23	24	Robusta	luis robles	Ecuador	robustasa	Lavado 1	our own lab	NaN
24	25	Robusta	luis robles	Ecuador	robustasa	Lavado 3	own laboratory	NaN
25	26	Robusta	james moore	United States	fazenda cazengo	NaN	cafe cazengo	NaN
26	27	Robusta	cafe politico	India	NaN	NaN	NaN	14-1118-2014-0087
27	28	Robusta	cafe politico	Vietnam	NaN	NaN	NaN	NaN

28 rows × 44 columns

This read in the data and printed it out because it is the last line on the cell. If we do something else after, it will read it in, but not print it out.

In order to use it, we save the output to a variable.

```
coffee_df = pd.read_csv(coffee_data_url)
```

we can look at it again using the jupyter display

```
coffee_df
```

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number
0	1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ankole coffee producers	0
1	2	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	25	sethuraman estate	14/1148/2017/21
2	3	Robusta	andrew hetzel	India	sethuraman estate	NaN	NaN	0000
3	4	Robusta	ugacof	Uganda	ugacof project area	NaN	ugacof	0
4	5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katuka development trust	0
5	6	Robusta	andrew hetzel	India	NaN	NaN	(self)	NaN
6	7	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN
7	8	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	7	sethuraman estate	14/1148/2017/18
8	9	Robusta	nishant gurjer	India	sethuraman estate	RKR	sethuraman estate	14/1148/2016/17
9	10	Robusta	ugacof	Uganda	ishaka	NaN	nsubuga umar	0
10	11	Robusta	ugacof	Uganda	ugacof project area	NaN	ugacof	0
11	12	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	RC AB	sethuraman estate	14/1148/2016/12
12	13	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN
13	14	Robusta	kasozi coffee farmers association	Uganda	kasozi coffee farmers	NaN	NaN	0
14	15	Robusta	ankole coffee producers coop	Uganda	kyangundu coop society	NaN	ankole coffee producers coop union ltd	0
15	16	Robusta	andrew hetzel	India	sethuraman estate	NaN	NaN	0000
16	17	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuraman estates	NaN

Skip to main content

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number
17	18	Robusta	kawacom uganda ltd	Uganda	bushenyi	NaN	kawacom	0
18	19	Robusta	nitubaasa ltd	Uganda	kigezi coffee farmers association	NaN	nitubaasa	0
19	20	Robusta	mannya coffee project	Uganda	mannya coffee project	NaN	mannya coffee project	0
20	21	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN
21	22	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuraman estates	NaN
22	23	Robusta	andrew hetzel	United States	sethuraman estates	NaN	sethuraman estates	NaN
23	24	Robusta	luis robles	Ecuador	robustasa	Lavado 1	our own lab	NaN
24	25	Robusta	luis robles	Ecuador	robustasa	Lavado 3	own laboratory	NaN
25	26	Robusta	james moore	United States	fazenda cazengo	NaN	cafe cazengo	NaN
26	27	Robusta	cafe politico	India	NaN	NaN	NaN	14-1118-2014-0087
27	28	Robusta	cafe politico	Vietnam	NaN	NaN	NaN	NaN

28 rows × 44 columns

Next we examine the type

```
type(coffee_df)
```

```
pandas.core.frame.DataFrame
```

This is a new type provided by the `pandas` library, called a dataframe

We can also exmaine its parts. It consists of several; first the column headings

```
coffee_df.columns
```

Skip to main content

```
Index(['Unnamed: 0', 'Species', 'Owner', 'Country.of.Origin', 'Farm.Name',
       'Lot.Number', 'Mill', 'ICO.Number', 'Company', 'Altitude', 'Region',
       'Producer', 'Number.of.Bags', 'Bag.Weight', 'In.Country.Partner',
       'Harvest.Year', 'Grading.Date', 'Owner.1', 'Variety',
       'Processing.Method', 'Fragrance...Aroma', 'Flavor', 'Aftertaste',
       'Salt...Acid', 'Bitter...Sweet', 'Mouthfeel', 'Uniform.Cup',
       'Clean.Cup', 'Balance', 'Cupper.Points', 'Total.Cup.Points', 'Moisture',
       'Category.One.Defects', 'Quakers', 'Color', 'Category.Two.Defects',
       'Expiration', 'Certification.Body', 'Certification.Address',
       'Certification.Contact', 'unit_of_measurement', 'altitude_low_meters',
       'altitude_high_meters', 'altitude_mean_meters'],
      dtype='object')
```

These are a special type called Index that is also provided by pandas.

It also tells us that the actual headings are of `dtype` `object`. `object` is used for strings or columns with mixed types

the **`dtype`** is slightly different from base Python types and is how pandas classifies but roughly is the same idea as a type.

```
type(coffee_df.columns)
```

```
pandas.core.indexes.base.Index
```

It also has an index (first column, visually) but it is special because this is how you can index the data.

```
coffee_df.index
```

```
RangeIndex(start=0, stop=28, step=1)
```

Right now this is an autogenerated index, but we can also use the `index_col` parameter to set that up front.

```
coffee_df = pd.read_csv(coffee_data_url,index_col=0)
coffee_df
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company
1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ankole coffee producers	0	ankole coffee producers coop
2	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	25	sethuraman estate	14/1148/2017/21	kaapi royale
3	Robusta	andrew hetzel	India	sethuraman estate	NaN	NaN	0000	sethuraman estate
4	Robusta	ugacof	Uganda	ugacof project area	NaN	ugacof	0	ugacof ltd
5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katuka development trust	0	katuka development trust ltd
6	Robusta	andrew hetzel	India	NaN	NaN	(self)	NaN	cafemakers, llc
7	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN	cafemakers
8	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	7	sethuraman estate	14/1148/2017/18	kaapi royale
9	Robusta	nishant gurjer	India	sethuraman estate	RKR	sethuraman estate	14/1148/2016/17	kaapi royale
10	Robusta	ugacof	Uganda	ishaka	NaN	nsubuga umar	0	ugacof ltd
11	Robusta	ugacof	Uganda	ugacof project area	NaN	ugacof	0	ugacof ltd
12	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	RC AB	sethuraman estate	14/1148/2016/12	kaapi royale
13	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN	cafemakers
14	Robusta	kasozi coffee farmers association	Uganda	kasozi coffee farmers	NaN	NaN	0	kasozi coffee farmers association
15	Robusta	ankole coffee producers coop	Uganda	kyangundu coop society	NaN	ankole coffee producers coop union ltd	0	ankole coffee producers coop
16	Robusta	andrew hetzel	India	sethuraman estate	NaN	NaN	0000	sethuraman estate
17	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuraman estates	NaN	cafemakers, llc
		kawacom						kawacom

Skip to main content

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company
19	Robusta	nitubaasa ltd	Uganda	kigezi coffee farmers association	NaN	nitubaasa	0	nitubaasa ltd
20	Robusta	mannya coffee project	Uganda	mannya coffee project	NaN	mannya coffee project	0	mannya coffee project
21	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN	cafemakers
22	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuraman estates	NaN	cafemakers, llc
23	Robusta	andrew hetzel	United States	sethuraman estates	NaN	sethuraman estates	NaN	cafemakers, llc
24	Robusta	luis robles	Ecuador	robustasa	Lavado 1	our own lab	NaN	robustasa
25	Robusta	luis robles	Ecuador	robustasa	Lavado 3	own laboratory	NaN	robustasa
26	Robusta	james moore	United States	fazenda cazengo	NaN	cafe cazengo	NaN	global opportunity fund
27	Robusta	cafe politico	India	NaN	NaN	NaN	14-1118-2014-0087	cafe politico
28	Robusta	cafe politico	Vietnam	NaN	NaN	NaN	NaN	cafe politico

28 rows × 43 columns

```
coffee_df.index
```

```
Index([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19, 20, 21, 22, 23, 24, 25, 26, 27, 28],
      dtype='int64')
```

Now we see that it uses the actual first column as the index that is bolded.

We can look at the first 5 rows with `head`

```
coffee_df.head()
```

Skip to main content

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company
1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ankole coffee producers	0	ankole coffee producers coop
2	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	25	sethuraman estate	14/1148/2017/21	kaapi royale
3	Robusta	andrew hetzel	India	sethuraman estate	NaN	NaN	0000	sethuraman estate
4	Robusta	ugacof	Uganda	ugacof project area	NaN	ugacof	0	ugacof ltd
5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katuka development trust	0	katuka development trust ltd

5 rows × 43 columns

Try it yourself

How can you look at the first 3 or last 2 rows?

and the last 5 with `tail`

```
coffee_df.tail()
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company	Altitude	F
24	Robusta	luis robles	Ecuador	robustasa	Lavado 1	our own lab	NaN	robustasa	NaN	sa
25	Robusta	luis robles	Ecuador	robustasa	Lavado 3	own laboratory	NaN	robustasa	40	sa
26	Robusta	james moore	United States	fazenda cazengo	NaN	cafe cazengo	NaN	global opportunity fund	795 meters	k pr
27	Robusta	cafe politico	India	NaN	NaN	NaN	14-1118-2014-0087	cafe politico	NaN	
28	Robusta	cafe politico	Vietnam	NaN	NaN	NaN	NaN	cafe politico	NaN	

5 rows × 43 columns

Important

We did not do this step in class

the shape of a DataFrame is an attribute

Skip to main content

```
coffee_df.shape
```

```
(28, 43)
```

We can pick out columns by name.

```
coffee_df['Color']
```

```
1            Green
2              NaN
3            Green
4            Green
5            Green
6            Green
7            Green
8     Bluish-Green
9            Green
10           Green
11           Green
12           Green
13           Green
14           Green
15           Green
16           Green
17      Blue-Green
18           Green
19           Green
20           Green
21    Bluish-Green
22           Green
23           Green
24      Blue-Green
25      Blue-Green
26             NaN
27           Green
28             NaN
Name: Color, dtype: object
```

a single column is a new type, called Series

```
type(coffee_df['Color'])
```

```
pandas.core.series.Series
```

We can pick out rows using the `loc` accessor. It is a tricky concept because it is **indexing** so it uses square brackets `[]` but it uses a `.` like a method. This is a sort of atypical syntax, but we do not use it very often. We pick out single columns a lot, so that has a nice easy syntax like above, but this is rare, so it got the less elegant syntax.

```
coffee_df.loc[1]
```

```
Species                                                Robusta
Owner                                ankole coffee producers coop
Country.of.Origin                                        Uganda
Farm.Name                          kyangundu cooperative society
Lot.Number                                                 NaN
Mill                                     ankole coffee producers
ICO.Number                                                   0
Company                              ankole coffee producers coop
Altitude                                                  1488
Region                                      sheema south western
Producer                             Ankole coffee producers coop
Number.of.Bags                                             300
Bag.Weight                                              60 kg
In.Country.Partner            Uganda Coffee Development Authority
Harvest.Year                                              2013
Grading.Date                                     June 26th, 2014
Owner.1                              Ankole coffee producers coop
Variety                                                    NaN
Processing.Method                                          NaN
Fragrance...Aroma                                         7.83
Flavor                                                    8.08
Aftertaste                                                7.75
Salt...Acid                                               7.92
Bitter...Sweet                                             8.0
Mouthfeel                                                 8.25
Uniform.Cup                                               10.0
Clean.Cup                                                 10.0
Balance                                                   7.92
Cupper.Points                                              8.0
Total.Cup.Points                                         83.75
Moisture                                                  0.12
Category.One.Defects                                         0
Quakers                                                      0
Color                                                    Green
Category.Two.Defects                                         2
Expiration                                       June 26th, 2015
Certification.Body            Uganda Coffee Development Authority
Certification.Address    e36d0270932c3b657e96b7b0278dfd85dc0fe743
Certification.Contact    03077a1c6bac60e6f514691634a7f6eb5c85aae8
unit_of_measurement                                          m
altitude_low_meters                                     1488.0
altitude_high_meters                                    1488.0
altitude_mean_meters                                    1488.0
Name: 1, dtype: object
```

We can also slice in dataframes, just like in strings.

```
subset_df = coffee_df.loc[5:8]
subset_df
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company
5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katuka development trust	0	katuka development trust ltd
6	Robusta	andrew hetzel	India	NaN	NaN	(self)	NaN	cafemakers, llc
7	Robusta	andrew hetzel	India	sethuraman estates	NaN	NaN	NaN	cafemakers
8	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	7	sethuraman estate	14/1148/2017/18	kaapi royale

4 rows × 43 columns

Now `loc[1]` will give a key error because there is no `1` in the index.

```
subset_df.loc[1]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/indexes/base.py:3653, in
   3652 try:
-> 3653     return self._engine.get_loc(casted_key)
   3654 except KeyError as err:

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/_libs/index.pyx:147, in pandas

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/_libs/index.pyx:176, in pandas

File pandas/_libs/hashtable_class_helper.pxi:2606, in pandas._libs.hashtable.Int64HashTable.get_item()

File pandas/_libs/hashtable_class_helper.pxi:2630, in pandas._libs.hashtable.Int64HashTable.get_item()

KeyError: 1

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[34], line 1
----> 1 subset_df.loc[1]

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/indexing.py:1103, in _Loc
   1100 axis = self.axis or 0
   1102 maybe_callable = com.apply_if_callable(key, self.obj)
-> 1103 return self._getitem_axis(maybe_callable, axis=axis)

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/indexing.py:1343, in _Loc
   1341 # fall thru to straight lookup
   1342 self._validate_key(key, axis)
-> 1343 return self._get_label(key, axis=axis)

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/indexing.py:1293, in _Loc
   1291 def _get_label(self, label, axis: AxisInt):
   1292     # GH#5567 this will fail if the label is not present in the axis.
-> 1293     return self.obj.xs(label, axis=axis)

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/generic.py:4095, in NDFra
   4093             new_index = index[loc]
   4094 else:
-> 4095     loc = index.get_loc(key)
   4097     if isinstance(loc, np.ndarray):
   4098         if loc.dtype == np.bool_:

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/indexes/base.py:3655, in
   3653     return self._engine.get_loc(casted_key)
   3654 except KeyError as err:
-> 3655     raise KeyError(key) from err
   3656 except TypeError:
   3657     # If we have a listlike key, _check_indexing_error will raise
   3658     #  InvalidIndexError. Otherwise we fall through and re-raise
   3659     #  the TypeError.
   3660     self._check_indexing_error(key)

KeyError: 1
```

the only values that will work in `loc` are the ones in the index:

```
subset_df.index
```

```
Index([5, 6, 7, 8], dtype='int64')
```

however, with `iloc` they are indexed by integer values starting with 0.

Skip to main content

```
subset_df.iloc[1]
```

```
Species                                                Robusta
Owner                                            andrew hetzel
Country.of.Origin                                        India
Farm.Name                                                  NaN
Lot.Number                                                 NaN
Mill                                                  (self)
ICO.Number                                                 NaN
Company                                        cafemakers, llc
Altitude                                                3000'
Region                                              chikmagalur
Producer                                       Sethuraman Estates
Number.of.Bags                                             200
Bag.Weight                                               1 kg
In.Country.Partner                  Specialty Coffee Association
Harvest.Year                                              2012
Grading.Date                                   February 29th, 2012
Owner.1                                              Andrew Hetzel
Variety                                                    NaN
Processing.Method                                          NaN
Fragrance...Aroma                                          8.0
Flavor                                                    7.92
Aftertaste                                                7.67
Salt...Acid                                                8.0
Bitter...Sweet                                            7.75
Mouthfeel                                                 7.75
Uniform.Cup                                               10.0
Clean.Cup                                                 10.0
Balance                                                   7.92
Cupper.Points                                             7.75
Total.Cup.Points                                         82.75
Moisture                                                   0.0
Category.One.Defects                                         0
Quakers                                                      0
Color                                                    Green
Category.Two.Defects                                         0
Expiration                                     February 28th, 2013
Certification.Body                  Specialty Coffee Association
Certification.Address    ff7c18ad303d4b603ac3f8cff7e611ffc735e720
Certification.Contact    352d0cf7f3e9be14dad7df644ad65efc27605ae2
unit_of_measurement                                            m
altitude_low_meters                                       3000.0
altitude_high_meters                                      3000.0
altitude_mean_meters                                      3000.0
Name: 6, dtype: object
```

2.9. Questions After Class

2.9.1. I think this something I need to figure but how do the localhost or just utilizing a url in VS Code? I was late to the class, I never got how to do the jupyter lab thing.

For this class, you need to use jupyter notebooks without extraneous metadata. If you use jupyter inside of vs code, it adds extraneous metadata that makes it hard to grade and VS code, in my experience, does not provide the most helpful autocomplete for Data Science.

Please see office hours to get help with it.

2.9.2. Is the Python we use in Jupyter lab notebooks any different from traditional Python?

the Python is mostly all the same. There are different python interpreters that have some slightly different behaviors, but mostly only in the display. As a matter of technicality, jupyter uses the ipython python as the kernel.

2.9.3. Does index just list all the rows?

the index is the *name* of the rows the same way that the column headers are the *name* of the columns.

2.9.4. How you copied the file url from github.

Click the raw button and then copy the URL from your browswer's url bar.

2.9.5. How did we change the index?

we changed the index from the inferred (figured out by pandas) `RangeIndex` to a column of the data by adding the `index_col=0` parameter to our `read_csv` call.

2.9.6. I would like to learn more about the panda commands

We will continue learning more pandas features for the next few weeks.

2.9.7. are we gonna have to use what we learned today in a bigger program in the future

Yes, these features we used today are the basis of all of the data analysis we will do all semester. However, we will not be writing "programs" the way you may have for other classes, we will be doing data analyses, which are more narrative.

2.9.8. How can I use Jupyter to clean data?

jupyter is a way to work with python code. We will learn what clean data looks like and more ways to manipulate dataframes to make it clean in two weeks.

2.9.9. Why is taking data from columns much more common than taking it from rows?

We set our data up so that each column is a varible. We often want to treat different variables differently, but do the same thing to all of the rows.

2.9.10. I was wondering more about the Index variable type and was also curious as to what that could be used for.

the `Index` type from `pandas` is a component of a `DataFrame`, we will use them implicitly whenever we work with a part of a dataframe and explicitly when we clean data.

Skip to main content

2.9.11. I know by convention we use the typical alias for importing libraries, but is it okay to use our own alias for our own private programs?

using nonstandard aliases is a bad habit to develop and I cannot endorse it. Technically the code will run, but in class it will be a style violation.

2.9.12. does the panda's data start indexing at 1 because 0 is where the table headers are located?

Indexing using `loc` started at 1 because the dataset had 1 there, in the second example it started at 5.

using `iloc` starts at 0.

2.9.13. In the dataset we loaded, I noticed that there were some zeros, which are nulls, I'm guessing we have to clean those out, and I was wondering how?

zeros are a value, nulls are encoded in different ways. We will learn how to deal will missing values in two weeks.

2.9.14. How often do data sets need to be cleaned/manipulated before proper analysis can be done?

Real data, will almost alwasy need to be fixed a little bit.

2.9.15. how does being able to view specific rows/columns help us make conclusions about data?

For example, maybe one column is the thing you are interested in, you may want to know on stats on the one column.

2.9.16. Are assignments always a certain level or can one assignment be done to be a level 1 or a level 2 assignment

Going forward, Assignments are always targeted at level 2. In class prismia questions will assess at level 1. An incomplete attempt at an assignment might be evaluated only at level 1, so that can be a way to make up for missed class and then you earn the level 2 in the next assignment that assess that skill.

2.9.17. will I be able to use data from an existing lab that I work in for certain assignment?

Yes, as long as the data is allowed to be shared. Please confirm with your PI.

2.9.18. When we submit to GitHub, do we need to do anything other than upload the file?

For your portfolio, no. For other assignments, there will be a step to do, and there will be instructions in the assignment.

2.9.19. Are we going to have to create our own datasets for any future assignments rather than downloading datasets form the Internet?

Assignment 2 you will build a dataset about datasets, but other than that you will mostly use datasets that you find online or that you have for another purpose.

2.9.20. How to better navigate github and not second guess my posts there

> **Warning**
>
> this will be added later.

3. DataFrames from other sources

Today we will:

- continue examining the dataframe object
- see more ways to load data
- make sure you are set up for assignment 2

3.1. Indexing reivew

```
topics = ['what is data science', 'jupyter', 'conditional','functions', 'lists', 'dictionaries','pandas' ]
topics[-1]
```

```
'pandas'
```

negative numbers count from the right

3.2. Reserve words

these are words you do not want to use for variable names

Python reserve words turn green:

```
print
```

```
<function print>
```

```
def
```

Skip to main content

```
  Cell In[3], line 1
    def
       ^
SyntaxError: invalid syntax
```

3.3. Built in iterable types

These are four different iterable constructions:

```
a = [char for char in 'abcde']
b =  {char:i for i, char in enumerate('abcde')}
c = ('a','b','c','d','e')
d = 'a b c d e'.split(' ')
```

We can see their types

```
type(a), type(b), type(c), type(d)
```

```
(list, dict, tuple, list)
```

Dictionaries are really useful because they consist of key, value pairs. This is really powerful and we will use it a lot to pass complex structures into functions.

```
b
```

```
{'a': 0, 'b': 1, 'c': 2, 'd': 3, 'e': 4}
```

```
a
```

```
['a', 'b', 'c', 'd', 'e']
```

Where we index lists with numbers

```
a[0]
```

```
'a'
```

we can access items, the values in a dictionary using square brackets and the keys

```
b['b']
```

```
1
```

Skip to main content

3.4. Building iterables quick with Comprehensions

- list comprehensions are **super** handy

we can make a list using a loop all in one line The constructions above for `a` and `b` are called list and dictionary **comprehensions** it is equivalent to using a loop, but a more concise way to build a list with a loop.

```
a_long = []
for char in 'abcde':
    a_long.append(char)
```

Notice that even in this for loop the lopo variable is a conceptually meaningful variable and we iterate over the items in an iterable type object. This is incontrast to creatinga loop variable that is an integer. This loop style is considered good pythonic strategy.

For more detail, see the Python docs section on looping strategies

```
a_long
```

```
['a', 'b', 'c', 'd', 'e']
```

Hint

> **Programming is a practice** the goal is not to memorize everything, but be exposed to enough that you remember what you can look up later

enumerate is a built in function that allows you to get both a number and an item for your use in a loop or comprehension. You can read the help below or the technical details in the official Python Docs

```
help(enumerate)
```

```
Help on class enumerate in module builtins:

class enumerate(object)
 |  enumerate(iterable, start=0)
 |
 |  Return an enumerate object.
 |
 |    iterable
 |      an object supporting iteration
 |
 |  The enumerate object yields pairs containing a count (from start, which
 |  defaults to zero) and a value yielded by the iterable argument.
 |
 |  enumerate is useful for obtaining an indexed list:
 |      (0, seq[0]), (1, seq[1]), (2, seq[2]), ...
 |
 |  Methods defined here:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __next__(self, /)
 |      Implement next(self).
 |
 |  __reduce__(...)
 |      Return state information for pickling.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
```

3.5. Read DataFrames from HTML

let's use `read_html` on the course communications page and then inspect what we get to figure it out

```
course_comms_url = 'https://rhodyprog4ds.github.io/BrownFall23/syllabus/communication.html'
```

we will first need out library

```
import pandas as pd
```

then we will read it in without saving it and look at the output to see what it looks like.

```
pd.read_html(course_comms_url)
```

```
[      Day      Time   Location       Host
 0  Monday  12pm-2pm       Zoom       Mark
 1  Monday     4-5pm       Zoom  Dr. Brown
 2  Friday     4-5pm  134 Tyler  Dr. Brown,
                                                 usage platform  \
 0                                         in class  prismia
 1                                         any time  prismia
 2               private questions to your assignment   github
 3         for general questions that can help others   github
 4  to share resources or ask general questions in...   github
 5       matters that don't fit into another category   e-mail

                            area  \
 0                          chat
 1           download transcript
 2      issue on assignment repo
 3       issue on course website
 4  discussion on community repo
 5        to brownsarahm@uri.edu

                                                  note
 0  outside of class time this is not monitored cl...
 1  use after class to get preliminary notes eg if...
 2                             eg bugs in your code"
 3  eg what the instructions of an assignment mean...
 4                    include links in your portfolio
 5  remember to include `[CSC310]` or `[DSP310]` (...  ]
```

now we will save it to a variable for future use.

```
comm_df_list = pd.read_html(course_comms_url)
```

we can check the type, it is a list as we noted from looking at the outpu.

```
type(comm_df_list)
```

```
list
```

and each item in the list is a DataFrame

```
type(comm_df_list[0])
```

```
pandas.core.frame.DataFrame
```

DataFrames also have a `shape` attribute, to tell us the number of rows and columns.

```
comm_df_list[0].shape
```

```
(3, 4)
```

```
achievements_url = 'https://rhodyprog4ds.github.io/BrownFall23/syllabus/achievements.html'
```

Skip to main content

This is a good job for a list comprehension.

```
shape_list_comp =[df.shape for df in pd.read_html(achievements_url)]
shape_list_comp
```

```
[(14, 3), (15, 5), (15, 15), (15, 6)]
```

Again, we can write this out as a for loop with append, but the comprehension is more concise.

```
shape_list = []
for df in pd.read_html(achievements_url):
    shape_list.append(df.shape)
```

in the comprehension structure the `[]` are what make it a list, they make anything a list

```
type([1,2,3])
```

```
list
```

3.6. More DataFrame Indexing

we'll go back to our coffee data

```
coffee_data_url = 'https://raw.githubusercontent.com/jldbc/coffee-quality-database/master/data/robusta_data_
```

```
coffee_df = pd.read_csv(coffee_data_url,index_col=0)
```

See again our shape

```
coffee_df.shape
```

```
(28, 43)
```

and the first few rows

```
coffee_df.head(1)
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company	Altitude	F
1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ankole coffee producers	0	ankole coffee producers coop	1488	s w

1 rows × 43 columns

Skip to main content

we can also see a random sample, not only the head and tail

```
coffee_df.sample(3)
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company	Alt
12	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	RC AB	sethuraman estate	14/1148/2016/12	kaapi royale	
14	Robusta	kasozi coffee farmers association	Uganda	kasozi coffee farmers	NaN	NaN	0	kasozi coffee farmers association	
25	Robusta	luis robles	Ecuador	robustasa	Lavado 3	own laboratory	NaN	robustasa	

3 rows × 43 columns

Hint

printing out the list of columns is a helpful way to get them to copy-paste for later selection to ensure no typos. In a polished notebook, you could then delete a cell like the one below, but it's really helpful while you are working

```
coffee_df.columns
```

```
Index(['Species', 'Owner', 'Country.of.Origin', 'Farm.Name', 'Lot.Number',
       'Mill', 'ICO.Number', 'Company', 'Altitude', 'Region', 'Producer',
       'Number.of.Bags', 'Bag.Weight', 'In.Country.Partner', 'Harvest.Year',
       'Grading.Date', 'Owner.1', 'Variety', 'Processing.Method',
       'Fragrance...Aroma', 'Flavor', 'Aftertaste', 'Salt...Acid',
       'Bitter...Sweet', 'Mouthfeel', 'Uniform.Cup', 'Clean.Cup', 'Balance',
       'Cupper.Points', 'Total.Cup.Points', 'Moisture', 'Category.One.Defects',
       'Quakers', 'Color', 'Category.Two.Defects', 'Expiration',
       'Certification.Body', 'Certification.Address', 'Certification.Contact',
       'unit_of_measurement', 'altitude_low_meters', 'altitude_high_meters',
       'altitude_mean_meters'],
      dtype='object')
```

We can subset columns by passing a list of multiple columns to use for indexing

```
columns_of_interest = ['Owner', 'Country.of.Origin']
coffee_df[columns_of_interest].head(1)
```

	Owner	Country.of.Origin
1	ankole coffee producers coop	Uganda

it has to be a list though, if we put them in one set of square brackets, it is a tuple and we get a `KeyError` because it looks for **one** column that has the name `'Owner', 'Country.of.Origin'`

```
coffee_df['Owner', 'Country.of.Origin']
```

Skip to main content

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/indexes/base.py:3653, in
   3652 try:
-> 3653     return self._engine.get_loc(casted_key)
   3654 except KeyError as err:

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/_libs/index.pyx:147, in pandas

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/_libs/index.pyx:176, in pandas

File pandas/_libs/hashtable_class_helper.pxi:7080, in pandas._libs.hashtable.PyObjectHashTable.get_item()

File pandas/_libs/hashtable_class_helper.pxi:7088, in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: ('Owner', 'Country.of.Origin')

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[31], line 1
----> 1 coffee_df['Owner', 'Country.of.Origin']

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/frame.py:3761, in DataFra
   3759 if self.columns.nlevels > 1:
   3760     return self._getitem_multilevel(key)
-> 3761 indexer = self.columns.get_loc(key)
   3762 if is_integer(indexer):
   3763     indexer = [indexer]

File /opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/indexes/base.py:3655, in
   3653     return self._engine.get_loc(casted_key)
   3654 except KeyError as err:
-> 3655     raise KeyError(key) from err
   3656 except TypeError:
   3657     # If we have a listlike key, _check_indexing_error will raise
   3658     #  InvalidIndexError. Otherwise we fall through and re-raise
   3659     #  the TypeError.
   3660     self._check_indexing_error(key)

KeyError: ('Owner', 'Country.of.Origin')
```

instead we can use 2 sets col square brackets if we do not want a separate variable

```
coffee_df[['Owner', 'Country.of.Origin']].head(1)
```

	Owner	Country.of.Origin
1	ankole coffee producers coop	Uganda

3.7. Subsetting by values

We can do boolean operators on a `pandas.Series` and it will do it automatically to every element

```
is_green = coffee_df['Color'] == 'Green'
is_green
```

```
1      True
2     False
3      True
4      True
5      True
6      True
7      True
8     False
9      True
10     True
11     True
12     True
13     True
14     True
15     True
16     True
17    False
18     True
19     True
20     True
21    False
22     True
23     True
24    False
25    False
26    False
27     True
28    False
Name: Color, dtype: bool
```

then we can look at the shape and see that it is the same shape as the column we selected.

```
is_green.shape, coffee_df['Color'].shape
```

```
((28,), (28,))
```

now we can use that to subset the rows

```
green_coffee_df = coffee_df[is_green]
green_coffee_df.head()
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company	Alti
1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ankole coffee producers	0	ankole coffee producers coop	
3	Robusta	andrew hetzel	India	sethuraman estate	NaN	NaN	0000	sethuraman estate	1(
4	Robusta	ugacof	Uganda	ugacof project area	NaN	ugacof	0	ugacof ltd	
5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katuka development trust	0	katuka development trust ltd	1
6	Robusta	andrew hetzel	India	NaN	NaN	(self)	NaN	cafemakers, llc	3

5 rows × 43 columns

and look at the shape to see

```
green_coffee_df.shape
```

```
(20, 43)
```

```
is_green.sum()
```

```
20
```

```
coffee_df.columns
```

```
Index(['Species', 'Owner', 'Country.of.Origin', 'Farm.Name', 'Lot.Number',
       'Mill', 'ICO.Number', 'Company', 'Altitude', 'Region', 'Producer',
       'Number.of.Bags', 'Bag.Weight', 'In.Country.Partner', 'Harvest.Year',
       'Grading.Date', 'Owner.1', 'Variety', 'Processing.Method',
       'Fragrance...Aroma', 'Flavor', 'Aftertaste', 'Salt...Acid',
       'Bitter...Sweet', 'Mouthfeel', 'Uniform.Cup', 'Clean.Cup', 'Balance',
       'Cupper.Points', 'Total.Cup.Points', 'Moisture', 'Category.One.Defects',
       'Quakers', 'Color', 'Category.Two.Defects', 'Expiration',
       'Certification.Body', 'Certification.Address', 'Certification.Contact',
       'unit_of_measurement', 'altitude_low_meters', 'altitude_high_meters',
       'altitude_mean_meters'],
      dtype='object')
```

3.8. Python has no switch

we use dictionaries in those kind of cases

Skip to main content

```
score_text = {False:'low',
    True:'high'}
```

Here we can switch from a list of true/false to high low

this gives true/false values for if the flavor is above or below 7

```
coffee_df['Flavor']>=7
```

```
1      True
2      True
3      True
4      True
5      True
6      True
7      True
8      True
9      True
10     True
11     True
12     True
13     True
14     True
15     True
16     True
17     True
18     True
19     True
20     True
21     True
22     True
23     True
24     True
25     True
26     True
27    False
28    False
Name: Flavor, dtype: bool
```

and this is high/low instead

```
[score_text[flavor_comp] for flavor_comp in coffee_df['Flavor']>=7]
```

```
['high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'high',
 'low',
 'low']
```

3.9. Keeping a clean notebook

we can put code in a python file and include it in our notebooks to use it

this can be useful if:

- you have a long hard to read thing that distracts from your other analysis
- you have a function you want to reuse a lot
- (unlikely in class) you need to make your own library!

I created a separate file calledn `example.py` and defined a variable in it like:

```
name ='sarah'
```

now I can import that and use it.

```
from example import name
```

```
name
```

```
'sarah'
```

3.10. Additional hints

Skip to main content

- using any pandas method is okay, including some we have not seen if it is a single method, for example the `select_dtypes` method docs
- Your task is partially to learn other IO methods, so the pandas docs IO page is a good resource

3.11. Questions After Class

> **Important**
>
> some questions are not answered below because they are explained in the notes above or they are too vague, you can come to office hours if you have a question that is not here or post a more detailed question on this repo or your assignment

3.11.1. what is pandas?

it is a Python library. Read more at the user guide

3.11.2. My question is how is the data frame being accessed from the url and how can I understand it more clearly?

`pd.read_` functions can do web requests and read data online and load it directly into memory. To understand in greater detail, I recommend the docs and then follow through the links through there to the level of depth that you want.

3.11.3. what does the shape of a dataframe do?

It is just information that we can do

3.11.4. Why do we need dictionaries to create new rows in the dataframes rather than operators?

We did not use the dictionary to create new rows, we used it to map values to other values. We will see this patther throughout the course.

3.11.5. how to figure out which dataframes from html are useful

we have to look at them.

3.11.6. How to download datasets

For your assignment, you can load directly with a URL

3.11.7. Is the sum() method only counting true values, and if so, is it simply treating them as 1?

Skip to main content

```
int(True), int(False)
```

```
(1, 0)
```

3.11.8. Why does the thing that happens right before a for in loop apply to all of the values? I think I know but just to be sure

in a list comprehension the part before the for is like the loop body, see above where I defined `a_long` and compare it to the defintion of `a`

3.11.9. I would like to learn more about dictionaries

I recommend starting in the python language docs section on dictionaries they are a very powerful structure and the text there is technical, but there are plenty of links. It is really good practice to get good at parsing through technical docs like this.

4. Exploratory Data Analysis

Now we get to start actual data science!

4.1. First, a note on assignments

4.1.1. the goal

- I am not looking for “an answer”
- You should not be either
- I am looking for evidence that you **understand** the material (thus far including prereqs)
- You should be trying to **understand** material

This means that in office hours, I am going to:

- ask you questions to help you think about the problem and the material
- help direct your attention to the right part of the error message to figure out what is wrong

4.1.2. Getting help

Sending me a screenshot is almost guaranteed to *not* get you a help. Not because I do not want to, but because I literally do not have the information to get you an answer.

Typically when someone do not know how to fix something from the error message, it is because they are reading the wrong part of the error message or looking at the wrong part of the code trying to find the problem.

This means they end up screenshotting that wrong thing, so I literally **cannot** tell what is wrong from the screenshot.

Skip to main content

I am not being stubborn, I just literally cannot tell what is wrong because I do not have enough information. Debugging code requires context, if you deprive me that, then I cannot help.

To get asynchronous help:

- upload your whole notebook, errors and all
- create an issue on that repo

4.2. This week: Exploratory Data Analysis

- How to summarize data
- Interpretting summaries
- Visualizing data
- interpretting summaries

4.2.1. Summarizing and Visualizing Data are **very** important

- People cannot interpret high dimensional or large samples quickly
- Important in EDA to help you make decisions about the rest of your analysis
- Important in how you report your results
- Summaries are similar calculations to performance metrics we will see later
- visualizations are often essential in debugging models

THEREFORE

- You have a lot of chances to earn summarize and visualize
- we will be picky when we assess if you earned them or not

```
import pandas as pd
```

```
coffee_data_url = 'https://raw.githubusercontent.com/jldbc/coffee-quality-database/master/data/robusta_data_
coffee_df = pd.read_csv(coffee_data_url, index_col=0)
```

```
coffee_df.head(1)
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company	Altitude	F
1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ankole coffee producers	0	ankole coffee producers coop	1488	s w

1 rows × 43 columns

4.3. Describing a Dataset

Skip to main content

So far, we've loaded data in a few different ways and then we've examined DataFrames as a data structure, looking at what different attributes they have and what some of the methods are, and how to get data into them.

We can also get more structural information with the `info`

```
coffee_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 28 entries, 1 to 28
Data columns (total 43 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Species                28 non-null     object
 1   Owner                  28 non-null     object
 2   Country.of.Origin      28 non-null     object
 3   Farm.Name              25 non-null     object
 4   Lot.Number             6 non-null      object
 5   Mill                   20 non-null     object
 6   ICO.Number             17 non-null     object
 7   Company                28 non-null     object
 8   Altitude               25 non-null     object
 9   Region                 26 non-null     object
 10  Producer               26 non-null     object
 11  Number.of.Bags         28 non-null     int64
 12  Bag.Weight             28 non-null     object
 13  In.Country.Partner     28 non-null     object
 14  Harvest.Year           28 non-null     int64
 15  Grading.Date           28 non-null     object
 16  Owner.1                28 non-null     object
 17  Variety                3 non-null      object
 18  Processing.Method      10 non-null     object
 19  Fragrance...Aroma      28 non-null     float64
 20  Flavor                 28 non-null     float64
 21  Aftertaste             28 non-null     float64
 22  Salt...Acid            28 non-null     float64
 23  Bitter...Sweet         28 non-null     float64
 24  Mouthfeel              28 non-null     float64
 25  Uniform.Cup            28 non-null     float64
 26  Clean.Cup              28 non-null     float64
 27  Balance                28 non-null     float64
 28  Cupper.Points          28 non-null     float64
 29  Total.Cup.Points       28 non-null     float64
 30  Moisture               28 non-null     float64
 31  Category.One.Defects   28 non-null     int64
 32  Quakers                28 non-null     int64
 33  Color                  25 non-null     object
 34  Category.Two.Defects   28 non-null     int64
 35  Expiration             28 non-null     object
 36  Certification.Body     28 non-null     object
 37  Certification.Address  28 non-null     object
 38  Certification.Contact  28 non-null     object
 39  unit_of_measurement    28 non-null     object
 40  altitude_low_meters    25 non-null     float64
 41  altitude_high_meters   25 non-null     float64
 42  altitude_mean_meters   25 non-null     float64
dtypes: float64(15), int64(5), object(23)
memory usage: 9.6+ KB
```

Now, we can actually start to analyze the data itself.

The `describe` method provides us with a set of summary statistics that broadly

```
coffee_df.describe()
```

Skip to main content

	Number.of.Bags	Harvest.Year	Fragrance...Aroma	Flavor	Aftertaste	Salt...Acid	Bitter...Sweet	Mouthf
count	28.000000	28.000000	28.000000	28.000000	28.000000	28.000000	28.000000	28.0000
mean	168.000000	2013.964286	7.702500	7.630714	7.559643	7.657143	7.675714	7.5067
std	143.226317	1.346660	0.296156	0.303656	0.342469	0.261773	0.317063	0.7251
min	1.000000	2012.000000	6.750000	6.670000	6.500000	6.830000	6.670000	5.0800
25%	1.000000	2013.000000	7.580000	7.560000	7.397500	7.560000	7.580000	7.5000
50%	170.000000	2014.000000	7.670000	7.710000	7.670000	7.710000	7.750000	7.6700
75%	320.000000	2015.000000	7.920000	7.830000	7.770000	7.830000	7.830000	7.8300
max	320.000000	2017.000000	8.330000	8.080000	7.920000	8.000000	8.420000	8.2500

From this, we can draw several conclusions. FOr example straightforward ones like:

- the smallest number of bags rated is 1 and at least 25% of the coffees rates only had 1 bag
- the first ratings included were 2012 and last in 2017 (min & max)
- the mean Mouthfeel was 7.5
- Category One defects are not very common (the 75th% is 0)

Or more nuanced ones that compare across variables like

- the raters scored coffee higher on Uniformity.Cup and Clean.Cup than other scores (mean score; only on the ones that seem to have a scale of up to 8/10)
- the coffee varied more in Mouthfeel and Balance that most other scores (the std; only on the ones that seem to have a scale of of up to 8/10)
- there are 3 ratings with no altitude (count of other variables is 28; alt is 25)

And these all give us a sense of the values and the distribution or spread fo the data in each column.

4.3.1. Understanding Quantiles

The 50% has another more common name: the median. It means 50% of the data are lower (and higher) than this value.

4.3.2. Individual variable

We can use the descriptive statistics on individual columns as well.

```
coffee_df['Balance'].descirbe()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
/tmp/ipykernel_2042/957071290.py in ?()
----> 1 coffee_df['Balance'].descirbe()

/opt/hostedtoolcache/Python/3.8.18/x64/lib/python3.8/site-packages/pandas/core/generic.py in ?(self, name)
   5985             and name not in self._accessors
   5986             and self._info_axis._can_hold_identifiers_and_holds_name(name)
   5987         ):
   5988             return self[name]
-> 5989         return object.__getattribute__(self, name)

AttributeError: 'Series' object has no attribute 'descirbe'
```

This is an `AttributeError` because there is no method or property of a `pd.Series` that is named `'descirbe'` that tells me it is a typo.

```
coffee_df['Balance'].describe()
```

```
count    28.000000
mean      7.541786
std       0.526076
min       5.250000
25%       7.500000
50%       7.670000
75%       7.830000
max       8.000000
Name: Balance, dtype: float64
```

4.4. Individual statistics

We can also extract each of the statistics that the `describe` method calculates individually, by name.

```
coffee_df.mean(numeric_only=True)
```

```
Number.of.Bags          168.000000
Harvest.Year           2013.964286
Fragrance...Aroma         7.702500
Flavor                    7.630714
Aftertaste                7.559643
Salt...Acid               7.657143
Bitter...Sweet            7.675714
Mouthfeel                 7.506786
Uniform.Cup               9.904286
Clean.Cup                 9.928214
Balance                   7.541786
Cupper.Points             7.761429
Total.Cup.Points         80.868929
Moisture                  0.065714
Category.One.Defects      2.964286
Quakers                   0.000000
Category.Two.Defects      1.892857
altitude_low_meters    1367.600000
altitude_high_meters   1387.600000
altitude_mean_meters   1377.600000
dtype: float64
```

Skip to main content

```
Number.of.Bags            1.00
Harvest.Year           2012.00
Fragrance...Aroma         6.75
Flavor                    6.67
Aftertaste                6.50
Salt...Acid               6.83
Bitter...Sweet            6.67
Mouthfeel                 5.08
Uniform.Cup               9.33
Clean.Cup                 9.33
Balance                   5.25
Cupper.Points             6.92
Total.Cup.Points         73.75
Moisture                  0.00
Category.One.Defects      0.00
Quakers                   0.00
Category.Two.Defects      0.00
altitude_low_meters      40.00
altitude_high_meters     40.00
altitude_mean_meters     40.00
dtype: float64
```

The quantiles are tricky, we cannot just `.25%()` to get the 25% percentile, we have to use the `quantile` method and pass it a value between 0 and 1.

```
coffee_df['Flavor'].quantile(.8)
```

```
7.83
```

```
coffee_df['Aftertaste'].mean()
```

```
7.559642857142856
```

4.5. Working with categorical data

There are different columns in the describe than the the whole dataset:

```
coffee_df.columns
```

```
Index(['Species', 'Owner', 'Country.of.Origin', 'Farm.Name', 'Lot.Number',
       'Mill', 'ICO.Number', 'Company', 'Altitude', 'Region', 'Producer',
       'Number.of.Bags', 'Bag.Weight', 'In.Country.Partner', 'Harvest.Year',
       'Grading.Date', 'Owner.1', 'Variety', 'Processing.Method',
       'Fragrance...Aroma', 'Flavor', 'Aftertaste', 'Salt...Acid',
       'Bitter...Sweet', 'Mouthfeel', 'Uniform.Cup', 'Clean.Cup', 'Balance',
       'Cupper.Points', 'Total.Cup.Points', 'Moisture', 'Category.One.Defects',
       'Quakers', 'Color', 'Category.Two.Defects', 'Expiration',
       'Certification.Body', 'Certification.Address', 'Certification.Contact',
       'unit_of_measurement', 'altitude_low_meters', 'altitude_high_meters',
       'altitude_mean_meters'],
      dtype='object')
```

So far, the stats above are only for numerical features.

Skip to main content

```
coffee_df['Color'].value_counts()
```

```
Color
Green           20
Blue-Green       3
Bluish-Green     2
Name: count, dtype: int64
```

Try it Yourself

Note `value_counts` does not count the `NaN` values, but `count` counts all of the not missing values and the shape of the DataFrame is the total number of rows. How can you get the number of missing Colors?

Describe only operates on the numerical columns, but we might want to know about the others. We can get the number of each value with `value_counts`

What is the most common country of origin?

```
coffee_df['Country.of.Origin'].value_counts()
```

```
Country.of.Origin
India            13
Uganda           10
United States     2
Ecuador           2
Vietnam           1
Name: count, dtype: int64
```

Value counts returns a pandas Series that has two parts: values and index

```
coffee_df['Country.of.Origin'].value_counts().values
```

```
array([13, 10,  2,  2,  1])
```

```
coffee_df['Country.of.Origin'].value_counts().index
```

```
Index(['India', 'Uganda', 'United States', 'Ecuador', 'Vietnam'], dtype='object', name='Country.of.Origin')
```

The `max` method takes the max of the values.

```
coffee_df['Country.of.Origin'].value_counts().max()
```

```
13
```

We can get the name of the most common country out of this Series using `idmax`

Skip to main content

```
coffee_df['Country.of.Origin'].value_counts().idxmax()
```

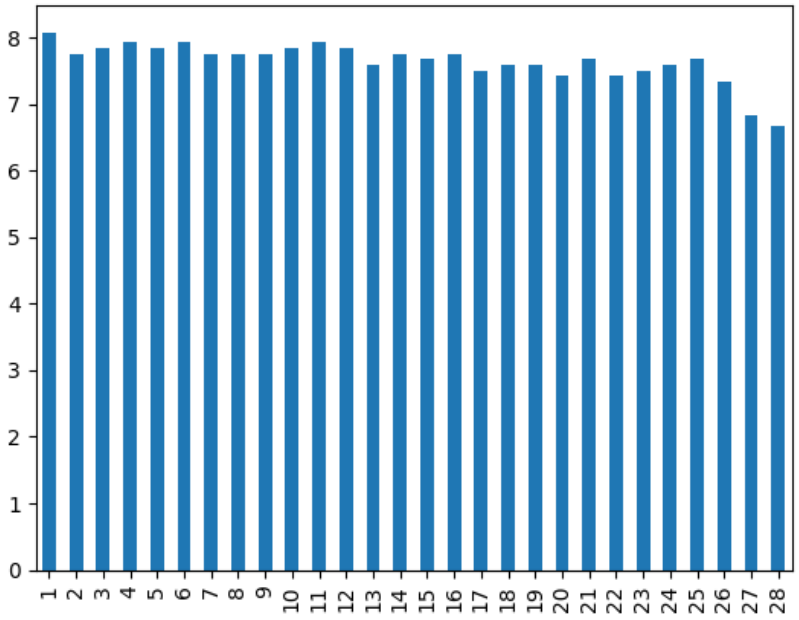
```
'India'
```

4.6. Which country scores highest on flavor?

Let's try to answer this with plots first

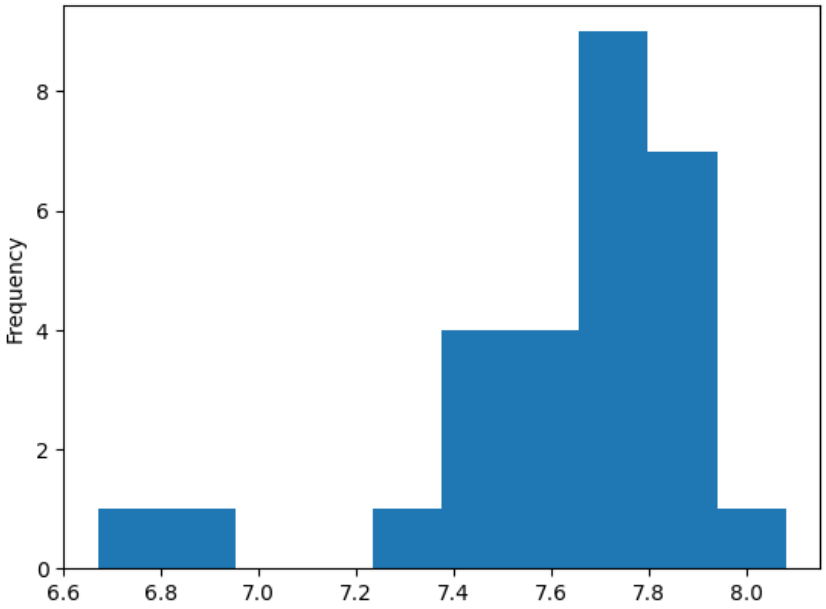
```
coffee_df['Flavor'].plot(kind='bar')
```

```
<Axes: >
```




```
coffee_df['Flavor'].plot(kind='hist')
```

```
<Axes: ylabel='Frequency'>
```



Seaborn give us *opinionated defaults*

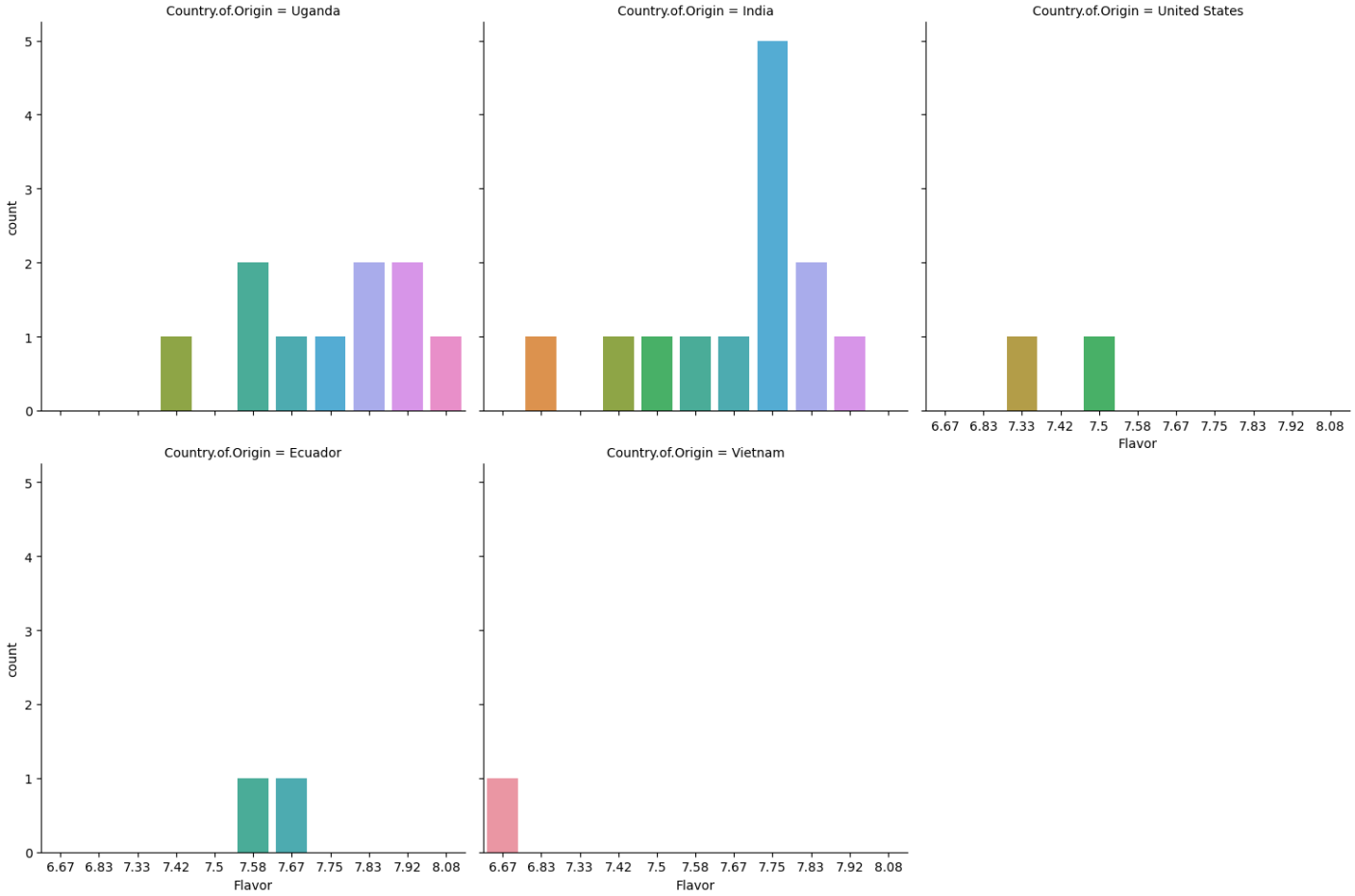
```
import seaborn as sns
```

seaborn's alias is `sns` as an inside joke among the developers to the character, Samual Norman Seaborn from West Wing they named the library after per their FAQ

So we can choose a typoe of plot and give it the whole dataframe and pass the variables to different parameters for them to be used in different ways.

```
sns.catplot(data =coffee_df, x='Flavor',kind ='count',col='Country.of.Origin',
            col_wrap=3)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f2147914be0>
```



4.6.1. Are any coffees more than one standard deviation above the average in `Flavor`?

```
coffee_df['Flavor'].mean() + coffee_df['Flavor'].std(), coffee_df['Flavor'].max()
```

```
(7.93437014076549, 8.08)
```

Yes, one is more than one standard deviation from the mean.

4.7. Questions after class

4.7.1. can pycharm access urls?

Probably, it should not block them and they are language features, but I do not use it to know.

4.7.2. can we do what we did today in pycharm?

Skip to main content

For this class, you need to submit notebook files, so pycharm will not work.

Pycharm is not typically used in datascience.

4.7.3. Are the bars on the Seaborn plot random colors, or do the colors have some meaning?

here, each different value is a different color. We wil control it more on Thursday

4.7.4. what types of jobs use what we learned today

Any data science role would use these skills daily.

4.7.5. Is there an equivalent for the index of the minimum value, such as idxmin()?

4.7.6. What is the std

standard deviation pandas docs on calculation. The wikipedia article on standard deviation is a good source on this.

4.7.7. Why can we not use matplotlib?

You *can* and you may for some customizaiton, but seaborn usually makes things faster to code and easier to read. I will not teach much matplotlib, but if you know it, up to date, not outdated, you may use it in some cases.

4.7.8. One question I have is what exactly the series object can be used for

A series is one column or one row of a dataframe.

4.7.9. how these lines of code can be used more efficiently in larger data sets?

For extremely large datasets if it's too much to plot, you might, for exaple, sample the dataset to only plot a subset.

Otherwise this will still work.

4.7.10. Can you change what the plot graphs rather than an index vs frequency?

yes, we can change the plot to whatever we want.

4.7.11. actually I would like to know like with our country/flavor example can they be put in the same plot but with different colors for the different countries to each country side by side for a given score

yes

4.7.12. Can assignment 3 can be release so we can pre-read it over.

Yes, it is posted now

4.8. Questions we'll answer in class on thurday

> **Important**
>
> These are **great** questions, I'm just not going to answer them here and then again in class on Thursday

- What other types of graphs could be used?
- Is there a way to use both x and y in sns without creating multiple charts?
- Using matplotlib, we never got to use it to class, and I want to know the difference between that and seanborn
- How to change the colors of the plots.
- What do we usually tend to look for in terms of patterns when trying to formulate questions about the data, or several pieces of data?

1. Assignment 1: Setup, Syllabus, and Review

Due: 2023-09-11

1.1. Evaluation

Eligible skills: (links to checklists)

- ★[1] python level 1 and level 2
- ★process[2] level 1

1.2. Related notes

> **Warning**
>
> the links below will not work until the relevant notes are posted, after class

- Welcome & What is Data Science

1.3. Instructions

> **Important**
>
> If you have trouble, check the GitHub FAQ on the left first

Skip to main content

1. Install required software from the Tools & Resource page (should have been done before the first class)
2. Create your portfolio, by accepting the assignment
3. Learn about your portfolio from the README file on your repository.
4. Follow instructions in the README to make your portfolio your own with information about yourself(not evaluated, but useful) and your own definition of data science (graded for **level 1 process**)
5. complete the `success.md`` file as per the instructions in the comments
6. Create a Jupyter notebook called `grading.ipynb` and write a function that computes a grade for this course, with the docstring below.
7. Upload the notebook to your repo directly on the main branch.
8. Add the line `  - file: grading` in your `_toc.yml` file.

Important

the syntax of the line added to your `_toc.yml` has to be exact

Warning

Do not merge your “Feedback” Pull Request

1.3.1. Docstring

```
    '''
    Computes a grade for CSC/DSP310 from numbers of achievements at each level

    Parameters:
    ------------
    num_level1 : int
      number of level 1 achievements earned
    num_level2 : int
      number of level 2 achievements earned
    num_level3 : int
      number of level 3 achievements earned

    Returns:
    --------
    letter_grade : string
      letter grade with possible modifier (+/-)
    '''
```

1.3.2. Sample tests

Here are some sample tests you could run to confirm that your function works correctly:

```
assert compute_grade(15,15,15) == 'A'

assert compute_grade(15,15,13) == 'A-'

assert compute_grade(15,14,14) == 'B-'

assert compute_grade(14,14,14) == 'C-'
```

Skip to main content

```
assert compute_grade(15,15,6) =='B+'
```

1.3.3. Notebook Checklist

- a Markdown cell with a heading
- your function called `compute_grade`
- three calls to your function that verify it returns the correct value for different number of badges that produce at three different letter grades.

1.3.4. Grading Notes:

- a basic function that uses conditionals in python will earn **level 1 python**
- to earn **level 2 python** use pythonic code to write a loop that tests your function's correctness, by iterating over a list or dictionary. Remember you will have many chances to earn level 2 achievement in python, so you do not need to do this step for this assignment if you are not sure how.

---

[1] skills will be marked like this on the first time they are eligible. There will also be a ⌛ on skills for the last assignment they are eligible

[2] process is a special skill. You'll earn level 1 in this assignment or a soon one and level two in either portfolio 1 or assignments 6-10, then level 3 in portfolio 2,3, or 4.

2. Assignment 2: Practicing Python and Accessing Data

Quick Facts

- due : 2023-09-18
- accept assignment

2.1. Objective & Evaluation

This assignment is an opportunity to earn level 1 and 2 achievements in `python` and `access` and begin working toward level 1 for `summarize`. You can also earn level 1 for `process`.

Eligible skills: (links to checklists)

- **first chance** access 1 and 2
- python 1 and 2
- summarize 1
- process 1

This assignment is an opportunity to earn level 1 and 2 achievements in `python` and `access` and begin working toward level 1 for `summarize`. You can also earn level 1 for `process`.

Skip to main content

2.2. Related notes

- Iterables and Pandas Data Frames
- DataFrames from other sources

2.3. Setting

Next week, we are going to learn about summarizing data. In this assignment, you are going to build a small dataset about datasets. In class next week, we will combine all of your datasets about datasets together in order to be able to answer questions like:

- how much total data did you all load
- how many students picked the same dataset?
- how many total rows of data did each student load?

2.4. Find Datasets

Find 3 datasets of interest to you that are provided in at least two different file formats. Choose datasets that are not too big, so that they do not take more than a few second to load. At least one dataset, must have non numerical (eg string or boolean) data in at least 1 column.

In your notebook, create a markdown cell for each dataset that includes:

- heading of the dataset's name
- a 1-2 sentence summary of what the dataset contains and why it was collected
- a "more info" link to where someone can learn about the dataset
- 1-2 questions you would like to answer with that dataset.

2.5. Store info about data for loading

Create a list of dictionaries in `datasets.py`, so that there is one dictionary for each dataset. Each dictionary should have the following keys:

Table 2.1 Meta data of the dictionaries

`url`	the full url of the dataset
`short_name`	a short name
`load_function`	(the actual function handle) what function should be used to load the data into a `pandas.DataFrame`.

Hint

See below for how you will use the dictionary as help for how you should construct it

Skip to main content

2.6. Make a dataset about your datasets

In a notebook called `dataset_of_datasets.ipynb`, import the list of dictionaries from the `datasets` module you created in the step above. Then iterate over the list of dictionaries, and:

1. load each dataset like `dataset_dict['load_function'](dataset_dict['url'])`
2. save it to a local csv using the short name you provided for the dataset as the file name, without writing the index column to the file.
3. record attributes about the dataset as in the table below in a list or dictionary of lists
4. Use that to create a DataFrame with columns that match the rows of the following table.

Table 2.2 Meta Data Description of the DataFrame to build

name	a short name for the dataset
source	a url to where you found the data
num_rows	number of rows in the dataset
num_columns	number of columns in the dataset
num_numerical	number of numerical variables in the dataset

2.7. Explore Your Datasets

In a second notebook file called `exploration.ipynb`:

For one dataset that includes nonnumerical data:

- read it in from your local csv using a relative path
- display the heading and the first 6 rows
- make a numpy array of only the numerical data and save it to a new variable (select these programmatically)
- was the format that the data was provided in a good format? why or why not?

For any other dataset:

- read it in from your local csv using a relative path
- display the heading with the last seven rows
- display the datatype for each column
- Are there any variables where pandas may have read in the data as a datatype that's not what you expect (eg a numerical column mistaken for strings)? If so, investigate and try to figure out why.

For the third dataset:

- read it in from your local csv using a relative path
- save every fifth row (5,10,15 ,…) of the data for two columns of your choice into a new DataFrame and display that

2.8. Exploring data files

Skip to main content

There are two files in the data folder, both can be read in with `read_csv` but need some options or fixing.

- try to read in the `german.data` file, what happens with the default settings? What option do you need to use to make it look right?
- try to read in the `.csv` file that's included in the template repository, use the error messages you get to try to fix the file manually (any text editor, including jupyter can edit a `.csv`), making notes about what changes you made in a markdown cell.

> **Hint**
>
> For the csv file in the template's data folder, in Jupyter Lab, it will not let you edit a .csv file, but you can chagne the file name to txt (in your code too) and then it will work.

2.9. Submission

This time you have to separately submit from posting your code to make grading easier.

1. Go to the actions tab
2. Click the action called "Prepare & Submit" in the left hand sidebar
3. click the run workflow button on the right hand side.
4. Cilck run workflow

> **Hint**
>
> see the github docs for screenshots of how to do these steps.

2.10. Thinking ahead

> **Important**
>
> This section is not required, but is intended to help you get started thinking about ideas for your portfolio. If you complete it, we'll give your feedback to help shape your ideas to get to level 3 achievements. If you want to focus only on level 2 at this moment in time, feel free to skip this part. You could also think about these after submitting the assignment. If you want, you could discuss these ideas in office hours.

1. When might you prefer one datatype over another?
2. How does PEP 8 standard code help you be collaborative?
3. Learn about Datasheets for Datasets and find some examples, (eg this google scholar result) How could something like this impact your work as a data scientist?

3. Assignment 3: Exploratory Data Analysis

__Due:2023-09-25 __

Important

You have the option to work with a partner. You must plan this in advance so that you have access to collaborate. If you did not find a partner in class and you would like one, try to find one on the class discussion. @brownsarahm if you do not get a reply.

3.1. Objective & Evaluation

This week your goal is to do a small exploratory data analysis for two datasets of your choice.

Eligible skills: (links to checklists)

- process 1
- access 1 and 2
- **first chance** summarize 1 and 2
- **first chance** visualize 1 and 2

3.2. Related notes

- Exploratory Data Analysis
-

3.3. Choose Datasets

Each Dataset must have at least three variables, but can have more. Both datasets must have multiple types of variables. These can be datasets you used last week, if they meet the criteria below.

3.3.1. Dataset 1 (d1)

must include at least:

- two continuous valued variables **and**
- one categorical variable.

Hint

a dataset from the UCI data repository that's for classification and has continuous features would work for this

3.3.2. Dataset 2 (d2)

must include at least:

- two categorical variables **and**
- one continuous valued variable

Use a separate notebook for each dataset, name them `dataset_01.ipynb` and `dataset_02.ipynb`.

For **each** dataset, in the corresponding notebook complete the following:

1. Load the data to a notebook as a `DataFrame` from url or local path, if local, include the data file in your repository.
2. Explore the dataset in a notebook enough to describe its structure use the heading `## Description`
    - shape
    - columns
    - variable types
    - overall summary statisics
3. Write a short description of what the data contains and what it could be used for
4. Include overall summary for the data and interpret what that means. This should include code that generates the statistical summary and sentences in English in a markdown cell with your conclusions and explanation of the statistical summary. Are there limitations in how to safely interpre the data that the summary helps you see? are the variables what you expect?
5. Ask and answer 3 questions by using and interpreting statistics and visualizations as appropriate. Include a heading for each question using a markdown cell and H2:`##`. Make sure your analyses meet the criteria in the check lists below. Use the checklists to think of what kinds of questions would use those type of analyses and help shape your questions. (if you have one really complex question that can cover the checklists below, fewer than 3 questions is okay)
6. Describe what, if anything might need to be done to clean or prepare this data for further analysis in a finale `## Future analysis` markdown cell in your notebook.

3.4.1. Question checklist

be sure that every question (all six, 3 per dataset) has:

- a heading
- at least 1 statistic or plot
- interpretation that answers the question

3.4.2. Dataset 1 Checklist

make sure that your `dataset_01.ipynb` has:

3.4.2.1. Question version

> **Warning**
>
> these 3 should be equivalent to what is in the code version below, see those to make this more concrete if it does not make sense

- One overview of the relationship of a categorical variable to many numerical variables
- One question about a categorical variable and one numerical variable
- One question about the relationship between 3 variables

- Overall summary statistics grouped by a categorical variable
- A single statistic grouped by a categorical variable
- at least one plot that uses 3 total variables
- a plot and summary table that convey the same information. This can be one statistic or many.

3.4.3. Dataset 2 Checklist

make sure that your `dataset_02.ipynb` has:

3.4.3.1. Code version

- overall summary statistics
- two individual summary statistics for one variable
- one summary statistic grouped by two categorical variables
- a figure with a grid of subplots that correspond to two categorical variables

3.4.3.2. Question version

> **Warning**
>
> these 3 should be equivalent to what is in the code version see those to make this more concrete if it does not make sense

- One question that is about overall trends across multiple variables. (the interpretation here is most important)
- One question that is about one variable's range or shape so that it requires to statistics to answer it.
- One question that is about how two categorical variables influence one numerical variable

> **Tip**
>
> Be sure to start early and use help hours to make sure you have a plan for all of these.

3.5. Peer Review

> **Note**
>
> This is optional, but if you do a review, you only need to do one analysis each.

With a partner (or group of 3 where person 1 review's 2 work, 2 reviews 3, and 3 reviews 1) read your partner's notebook and complete a peer review on their pull request. You can do peer review when you have done most of your analysis, and explanation, even if some parts of the code do not work.

In your review:

- Use inline comments to denote places that are confusing or if you see solutions to problems your classmate could not solve

3.5.1. Review Questions

1. Describe overall how it was to read the analysis overall to read. Was it easy? hard? cohesive? jumpy?
2. How did the data summaries help prepare you to read the rest of the analysis? What do you think might be missing?
3. Do the questions make sense based on the data? Are they interesting questions? What could improve the questions
4. Are the statistics and plots appropriate for the questions?
5. Are the interpretations complete, clear, and consistent with the statistics and plots?
6. What could be done to make the explanations more clear and complete?
7. What additional analysis might make the analysis more compelling and clear?

3.5.2. Response

Respond to your review either inline comments, replies, or by updating your analysis accordingly.

3.6. Think Ahead

Think Ahead

1. How could you make more customized summary tables?
2. Could you use any of the variables in this dataset to add more variables that would make interesting ways to apply split-apply-combine? (eg thresholding a continuous value to make a categorical value)

War

This s
intend
thinki
If you
feedb
get to
want
mome
part.

Portfolio

This section of the site has a set of portfolio prompts and this page has instructions for portfolio submissions.

Starting in week 3 it is recommended that you spend some time each week working on items for your portfolio, that way when it's time to submit you only have a little bit to add before submission.

The portfolio is your only chance to earn Level 3 achievements, however, if you have not earned a level 2 for any of the skills in a given check, you could earn level 2 then instead. The prompts provide a starting point, but remember that to earn achievements, you'll be evaluated by the rubric. You can see the full rubric for all portfolios in the syllabus. Your portfolio is also an opportunity to be creative, explore things, and answer your own questions that we haven't answered in class to dig deeper on the topics we're covering. Use the feedback you get on assignments to inspire your portfolio.

Each submission should include an introduction and a number of 'chapters'. The grade will be based on both that you demonstrate skills through your chapters that are inspired by the prompts and that your summary demonstrates that you *know* you learned the skills. See the formatting tips for advice on how to structure files.

On each chapter(for a file) of your portfolio, you should identify which skills by their keyword, you are applying.

You can view a (fake) example in this repository as a pdf or as a rendered website

Upcoming Checks

Skip to main content

Portfolio check 2 will assess the following *new* achievements in addition to an a chance to make up any that you have missed:

keyword	Level 3
python	reliable, efficient, pythonic code that consistently adheres to pep8
process	Compare different ways that data science can facilitate decision making
access	access data from both common and uncommon formats and identify best practices for formats in different contexts
construct	merge data that is not automatically aligned
summarize	Compute and interpret various summary statistics of subsets of data
visualize	generate complex plots with pandas and plotting libraries and customize with matplotlib or additional parameters
prepare	apply data reshaping, cleaning, and filtering manipulations reliably and correctly by assessing data as received
evaluate	Evaluate a model with multiple metrics and cross validation
classification	fit and apply classification models and select appropriate classification models for different contexts
regression	fit and explain regrularized or nonlinear regression
clustering	apply multiple clustering techniques, and interpret results

Formatting Tips

> **Warning**
>
> This is all based on you having accepted the portfolio assignment on github and having a cloned copy of the template. If you are not enrolled or the initial assignment has not been issued, you can view the template on GitHub

Your portfolio is a jupyter book. This means a few things:

- it uses myst markdown
- it will run and compile Jupyter notebooks

This page will cover a few basic tips.

Managing Files and version

You can either convert your ipynb files to earier to read locally or on GitHub.

The GitHub version means installing less locally, but means that after you push changes, you'll need to pull the changes that GitHub makes.

To manage with a precommit hook jupytext conversion

change your `.pre-commit-config.yaml` file to match the following:

Skip to main content

```
-   repo: https://github.com/mwouts/jupytext
    rev: v1.10.0  # CURRENT_TAG/COMMIT_HASH
    hooks:
    - id: jupytext
      args: [--from, ipynb, --to, myst]
```

Run Precommit over all the files to actually apply that script to your repo.

```
pre-commit install
pre-commit run --all-files
```

If you do `git status` now, you should have a `.md` file for each `ipynb` file that was in your repository, now add and commit those.

Now, each time you commit, it will run jupytext first.

To manage with a gh action jupytext conversion

create a file at `.github/workflows/jupytext.yml` and paste the following:

```
name: jupytext

# Only run this when the master branch changes
on:
  push:
    branches:
    - main
    # If your git repository has the Jupyter Book within some-subfolder next to
    # unrelated files, you can make this run only if a file within that specific
    # folder has been modified.
    #
    # paths:
    # - some-subfolder/**

# This job installs dependencies, build the book, and pushes it to `gh-pages`
jobs:
  jupytext:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v2

    # Install dependencies
    - name: Set up Python 3.7
      uses: actions/setup-python@v1
      with:
        python-version: 3.7

    - name: Install dependencies
      run: |
        pip install jupytext
    - name: convert
      run: |
          jupytext */*.ipynb --to myst
          jupytext *.ipynb --to myst
    - uses: EndBug/add-and-commit@v4 # You can change this to use a specific version
      with:
        # The arguments for the `git add` command (see the paragraph below for more info)
        # Default: '.'
        add: '.'

        # The name of the user that will be displayed as the author of the commit
        # Default: author of the commit that triggered the run
        author_name: Your Name
```

```
        # Default: author of the commit that triggered the run
        author_email: you@uri.edu

        # The local path to the directory where your repository is located. You should use actions/checkout t
        # Default: '.'
        cwd: '.'

        # Whether to use the --force option on `git add`, in order to bypass eventual gitignores
        # Default: false
        force: true

        # Whether to use the --signoff option on `git commit`
        # Default: false
        signoff: true

        # The message for the commit
        # Default: 'Commit from GitHub Actions'
        message: 'convert notebooks to md'

        # Name of the branch to use, if different from the one that triggered the workflow
        # Default: the branch that triggered the workflow (from GITHUB_REF)
        ref: 'main'

        # Name of the tag to add to the new commit (see the paragraph below for more info)
        # Default: ''
        tag: "v1.0.0"

      env:
        # This is necessary in order to push a commit to the repo
        GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }} # Leave this line unchanged
```

Organization

The summary of for the `part` or whole submission, should match the skills to the chapters. Which prompt you're addressing is not important, the prompts are a *starting point* not the end goal of your portfolio.

Data Files

Also note that for your portfolio to build, you will have to:

- include the data files in the repository and use a relative path OR
- load via url

using a full local path(eg that starts with `///file:`) **will not work** and will render your portfolio unreadable.

Structure of plain markdown

Use a heading like this:

```
# Heading of page
## Heading 2
### Heading 3
```

in the file and it will appear in the sidebar.

You can also make text *italic* or **bold** with either `*asterics*` or `__underscores__` with `_one for italic_` or `**two for`

File Naming

It is best practice to name files without spaces. Each `chapter` or file should have a descriptive file name (`with_no_spaces`) and descriptive title for it.

Syncing markdown and ipynb files

If you have the precommit hook working, git will call a script and convert your notebook files from the ipynb format (which is json like) to Myst Markdown, which is more plain text with some header information. The markdown format works better with version control, largely because it doesn't contain the outputs.

If you don't get the precommit hook working, but you do get jupytext installed, you can set each file to sync.

Adding annotations with formatting or margin notes

You can either install jupytext and convert locally or upload /push a notebook to your repository and let GitHub convert.
Then edit the .md file with a text editor of your choice. You can run by uploading if you don't have jupytext installed, or locally if you have installed jupytext or jupyterbook.

In your .md file use backticks to mark special content blocks

````
```{note}
Here is a note!
```
````

````
```{warning}
Here is a warning!
```
````

````
```{tip}
Here is a tip!
```
````

````
```{margin}
Here is a margin note!
```
````

For a complete list of options, see the **`sphinx-book-theme`** documentation.

Links

Markdown syntax for links

```
[text to show](path/or/url)
```

Things like the menus and links at the top are controlled as settings, in `_config.yml`. The following are some things that you might change in your configuration file.

Show errors and continue

To show errors and continue running the rest, add the following to your configuration file:

```
# Execution settings
execute:
  allow_errors              : true
```

Using additional packages

You'll have to add any additional packages you use (beyond pandas and seaborn) to the `requirements.txt` file in your portfolio.

Portfolio Check 1 Ideas

Remember you'll be graded against the [rubric] and the [achievement checklists], but these are ideas for the structure.

You can mix and match different formats to cover the skills collectively.

If your goal is, for example, a B+ (you need 5 level 3s) you could only do 1-2 skills per portfolio check (there are 4 checks).

Earning Level 3s

You could also submit a few shorter pieces that in total cover all of the skills. Some example formats:

Tutorial

Write a notebook that explains a concept related to a skill with examples in a real dataset and with visuals or a toy dataset (minimal number of columns rows)

Cheatsheet

Make a detailed reference with code outputs on a topic or a few topics.

Blog post

Write a blog post styled Notebook that compares or analyzes something, for example:

- how do different ways of loading data compare
- describe best practices you've learned and show why they're good with examples

Extension

If there were parts of your previous assignments that you thought were interesting and you want to work with those data more, you can. You need to do *more complex* analyses of them, but you can build off of what you already have done, especially for assignments 2, 3, and 5.

Correction & Reflection

If you had trouble with an assignment so far, you can revise what you submitted and resubmit it, with reflections and explanation of what you were confused about, what you tried initially, how you eventually figured it out, and explains the correct answer. Then go a little deeper in exploring the topic in that context to also earn level 3.

Practice Problems and Solutions

Based on the level 3 rubric descriptions, write practice problems that build off of the lecture notes. Include solutions and descriptions for each. These can be open ended or multiple choice questions with plausible distractors. A plausible distractor is an incorrect answer that represents a way that you think someone could misunderstand.

For example if the question is 37 + 15 = ?, MCQ with plausible distractors might be:

- 52 (correct)
- 412 (didn't carry the one, correctly: 7+5 = 12, 3+1 = 4)
- 42 (dropped the one 7+5 = 12, ones place is 2, 3+1 = 4)
- 43 (carried one into wrong column, 7 + 5 = 12, 1+2 = 3, 3+1 = 3)

Long single analysis

Collect data from multiple sources, prepare each for analysis, and merge them together then do some exploratory data analysis. Describe each step, interpret all outputs, and put the analysis in context of the Data Science Process.

This would be one long notebook that covers all of the skills.

Be sure to check the checklists for how level 3s are more complex than level 2s. I recommend using office hours to help get ideas if you are not sure how to extend your analysis.

Check 2 Ideas

For Check 2, all of the prompts from check 1 apply, plus the following additional prompts, since there are new skills.

If you have other ideas, you can also ask and those are likely posible.

Level 1 Achievement Catchup

To make up level 1 achievements, include a detailed introduction file to your portfolio and one of the following (per skill):

- minor extensions to what we did in class
- answers to problems from the notes

- psuedocode for one of the other prompts

Extend Assignment 7, 8, or 9

Assignments 7-9 help you think through what machine learning tasks are. Extend those ideas by adding additional experiments based on your own questions or the questions in your feedback.

Build a data set for Prediction

Build a dataset that works for prediction (classification, regression, or clustering) from other sources.

Learn a new model

Repeat what you did in 7, 8, or 9, with a different model.

Create datasets that fail

Create datasets that violate assumptions of a model we have learned. The sklearn data generators are a good place to start.

Process level 3

Process level 3 is a little different than most of the others. You may be able to work it into an analysis notebook, but likely, you'll need to do one of the following.

Data Science Pipeline Comparisons

Find two different sources that describe the data science pipeline or lifecycle. Write a blog style post that discusses their differences and hypothesizes about why they may be different? Are they for different audiences? Is one domain specific? How do they emphasize different modeling tasks? Include a Recommendation for when you think each one is better

Write a short story

Write a short story that explains the concepts of data science to demonstrate your understanding of process.

Media Review

Watch/listen/read to an episode of a high quality[1] podcast or other type of media and write a blog style summary and review. Highlight what you learned and how it relates to topics covered in class.

Approved Media:

- Pod of Asclepius, Fall Series: The Philosophy of Data Science
- Chapter 1 & 2 of Think like a Data Scientist in particular, if you think these would be helpful to assign as reading or teach from

- Algorithms of Oppression (book)
- Weapons of Math Destruction (book)
- Coded Bias (film, available on netflix & PBS)

---

[1] approved Dr. Brown by creating a pull request to add it to the list on this page that is successfully merged. To create a PR, use the suggest an edit button at the top of this page.

Check 3 Ideas

For Check 4, all of the prompts from check 1 &2 apply, plus the following additional prompts.

If you have other ideas, you can also ask and those are likely posible.

Organize your knowledge

Develop some sort of visual aid that demonstrates how you understand some aspect(s) of data science working. Think of this as something that future students could use to help them learning, so assume prior knowledge topics covered earlier than the one you are demonstrating.

This could be a concept map, a table that shows how you've traced how something works or any other sort of conceptual tol that helps convey your understanding.

Extend any assignment

Assignments 7-12 are most relevant because they leave room to extend and ask new questions.

If you both reflect on what you had trouble with and extend you could earn level 2 and 3.

Try alternative libraries/ tools

One option for workflow level 3 is to use other data science skills and reflect on how what we have learned so far helped you learn a new set of tool as an alternative way to do things.

Try feature engineering or representation learning

Try different transformations and see how they impact how well a model performs. This could be using `sklearn.feature_extraction` tools or trying different types of neural network layers at the beginning.

FAQ

This section will grow as questions are asked and new content is introduced to the site. You can submit questions:

- via e-mail to Dr. Brown (brownsarahm) or TA
- via Priamia chat during class

Syllabus and Grading FAQ

How much does assignment x, class participation, or a portfolio check weigh in my grade?

There is no specific weight for any activities, because your grade is based on earning achievements for the skills listed in the skills rubric.

However, if you do not submit (or earn no achievements from) assignments or portfolios, the maximum grade you can earn is a C. If you do not submit (or earn no achievements from) your portfolio, the maximum grade you can earn is a B.

What time are assignments due?

End of day. I could start grading at any time the next morning. If your work is not there when I start grading it will not be graded, but if it is, I won't check the time stamp.

Can I submit this assignment late if …?

Late assignments are not accepted, however, your grade is based on the skills, not the assignments. All skills are assessed in at least two assignments, so missing any one will not hurt your grade. If you need an accommodation because you cannot submit multiple assignments, contact Dr. Brown.

I don't understand my grade on this assignment

If you have questions about your grade, the best place to get feedback is to reply on the Feedback PR. Either reply directly to one of the inline comments, or the summary.

Be specific about what you think you should have earned and why.

Git and GitHub

I can't push to my repository, I get an error that updates were rejected

```
! [rejected] main -> main (fetch first)
error: failed to push some refs to <repository name>
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Your local version and github version are out of sync, you need to pull the changes from github to your local computer before you can push new changes there.

```
git pull
```

You'll probably have to resolve a merge conflict

The content I added to my portfolio isn't in the pdf

There was an error in the original `_toc.yml` file, change yours to match the following:

```
format: jb-book
root: intro
parts:
  - caption: About
    chapters:
    - file: about/index
    - file: about/grading
#  - caption: Check 1
#    chapters:
#    - file: submission_1_intro
```

uncomment the later lines and add any new files you add.

My command line says I cannot use a password

GitHub has strong rules about authentication You need to use SSH with a public/private key; HTTPS with a Personal Access Token or use the GitHub CLI auth

My .ipynb file isn't showing in the staging area or didn't push

.ipynb files are json that include all of the output, including tables as html and plots as svg, so, unlike plain code files, they don't play well with version control.

Your portfolio has `*/*.ipynb` in the `.gitignore` file, so that these files do not end up in your repository. Instead, you'll convert your notebooks to Myst Markdown with jupytext via a precommit hook.

Your portfolio has the code to do this already, what you should do is make sure that `pre-commit` is installed and then run `pre-commit install`
(see your portfolio's README.md file for more detail)

If this doesn't work, you can follow the alterntive in the porfolio readme.

If that doesn't work, and you have time before the deadline, create an issue to get help.

As a last resort, use the jupyter interface to download (File > Download as > …)your notebook as `.md` if avialable or `.py` if not and then move that file from your Downloads folder to your repository. We'll set up another workflow for future work

My portfolio won't compile

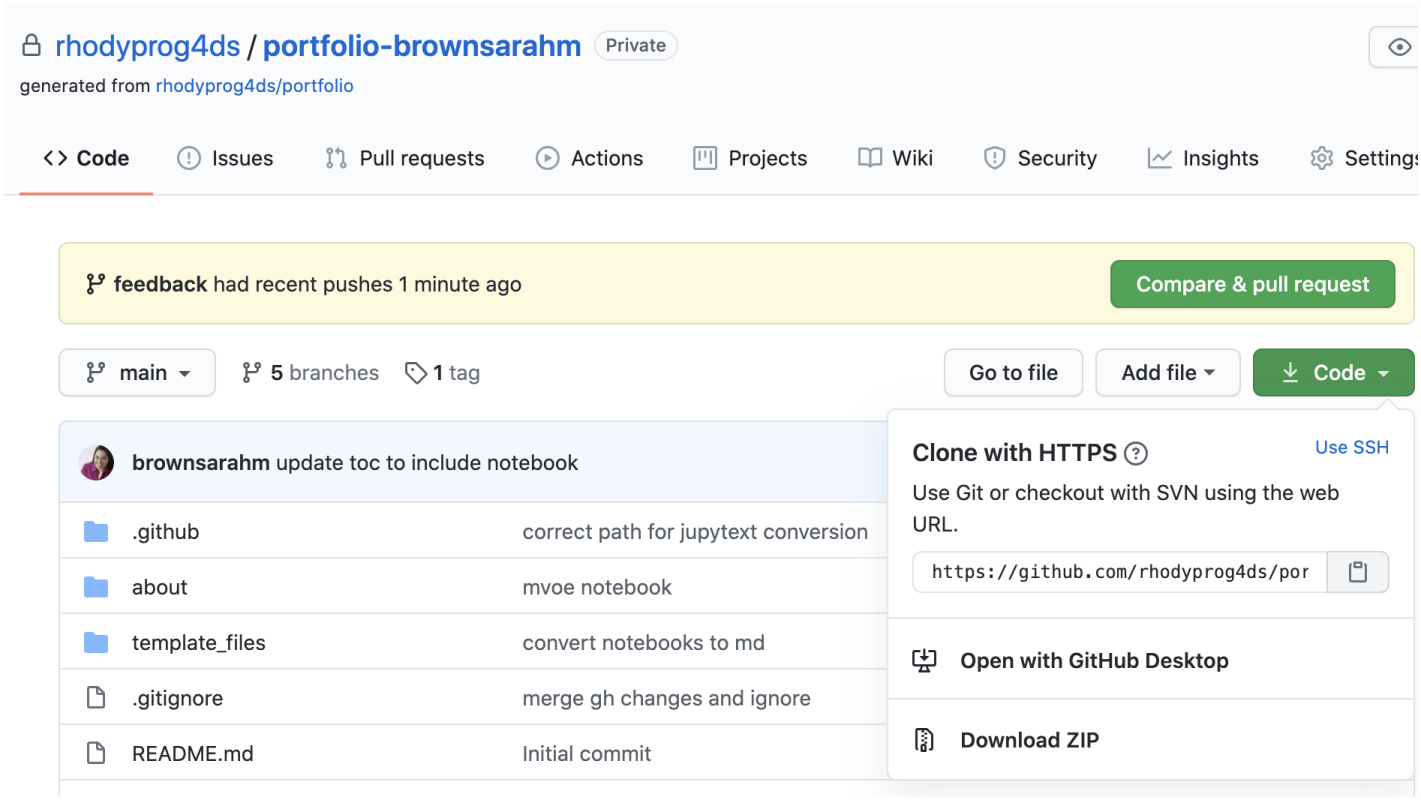
If there's an error your notebook it can't complete running. You can allow it to run if the error is on purpose by changing settings as

Help! I accidentally merged the Feedback Pull Request before my assignment was graded

That's ok. You can fix it.

You'll have to work offline and use GitHub in your browser together for this fix. The following instuctions will work in terminal on Mac or Linux or in GitBash for Windows. (see Programming Environment section on the tools page).

First get the url to clone your repository (unless you already have it cloned then skip ahead): on the main page for your repository, click the green "Code" button, then copy the url that's show



Next open a terminal or GitBash and type the following.

```
git clone
```

then past your url that you copied. It will look something like this, but the last part will be the current assignment repo and your username.

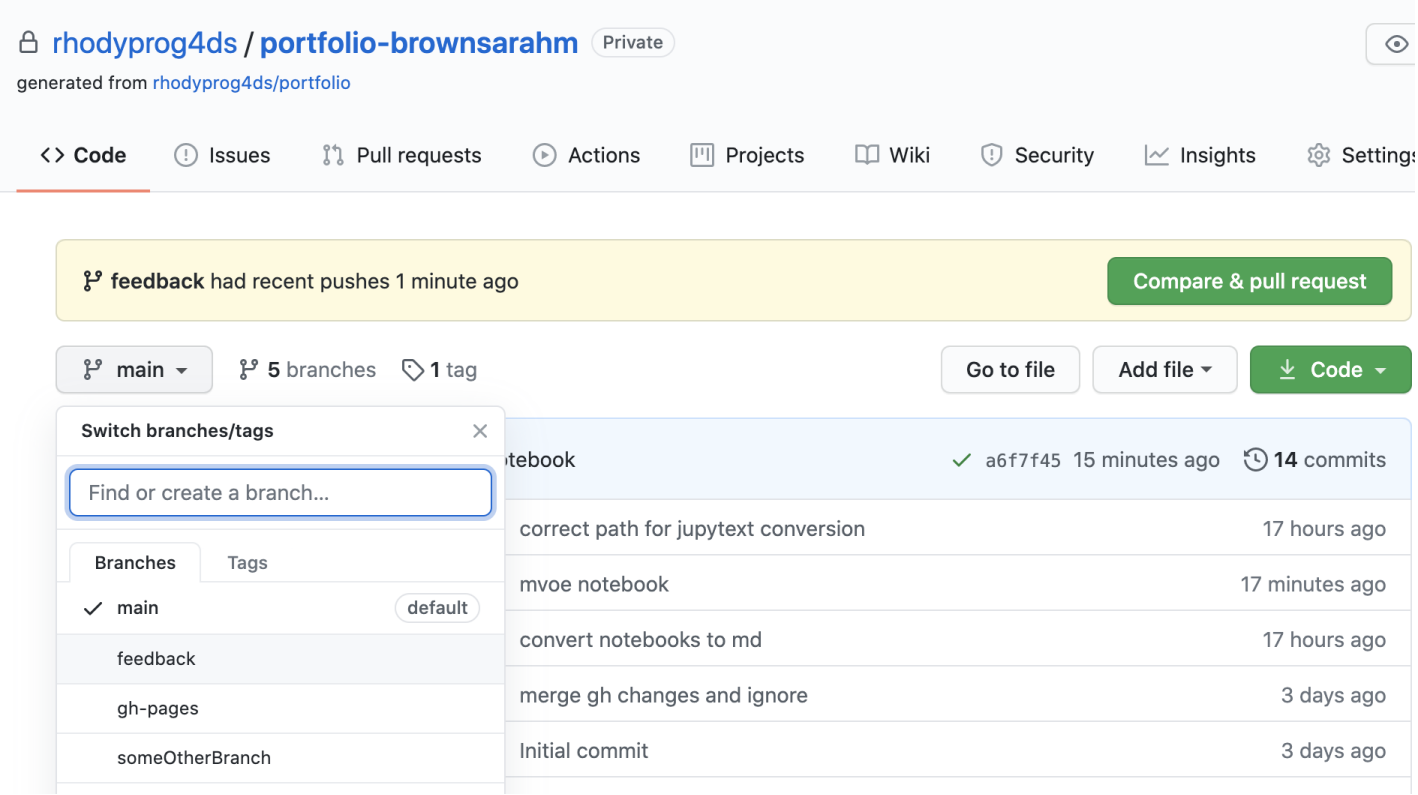
```
git clone https://github.com/rhodyprog4ds/portfolio-brownsarahm.git
```

When you merged the Feedback pull request you advanced the `feedback` branch, so we need to hard reset it back to before you did any work. To do this, first check it out, by navigating into the folder for your repository (created when you cloned above) and then checking it out, and making sure it's up to date with the `remote` (the copy on GitHub)

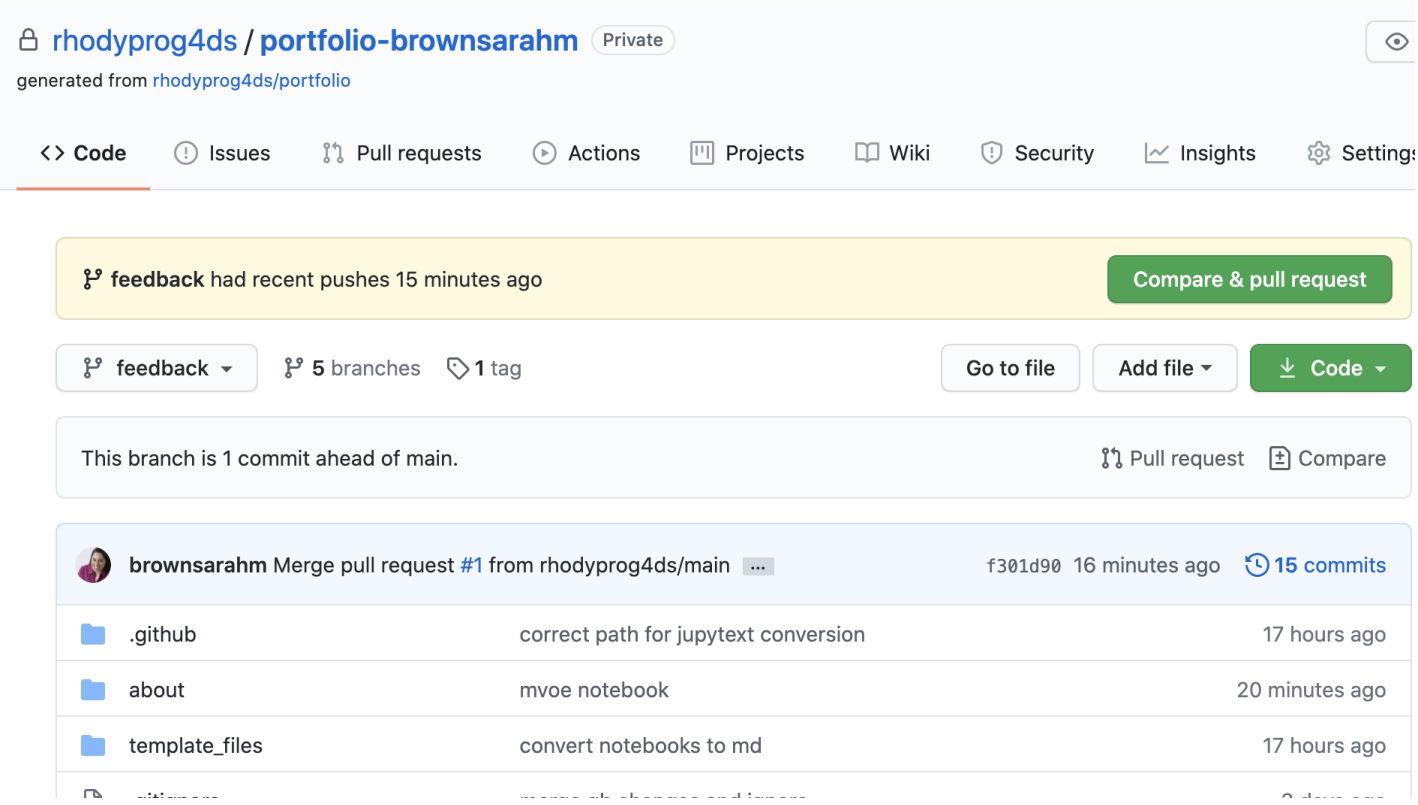
```
cd portfolio-brownsarahm
git checkout feedback
git pull
```

Skip to main content

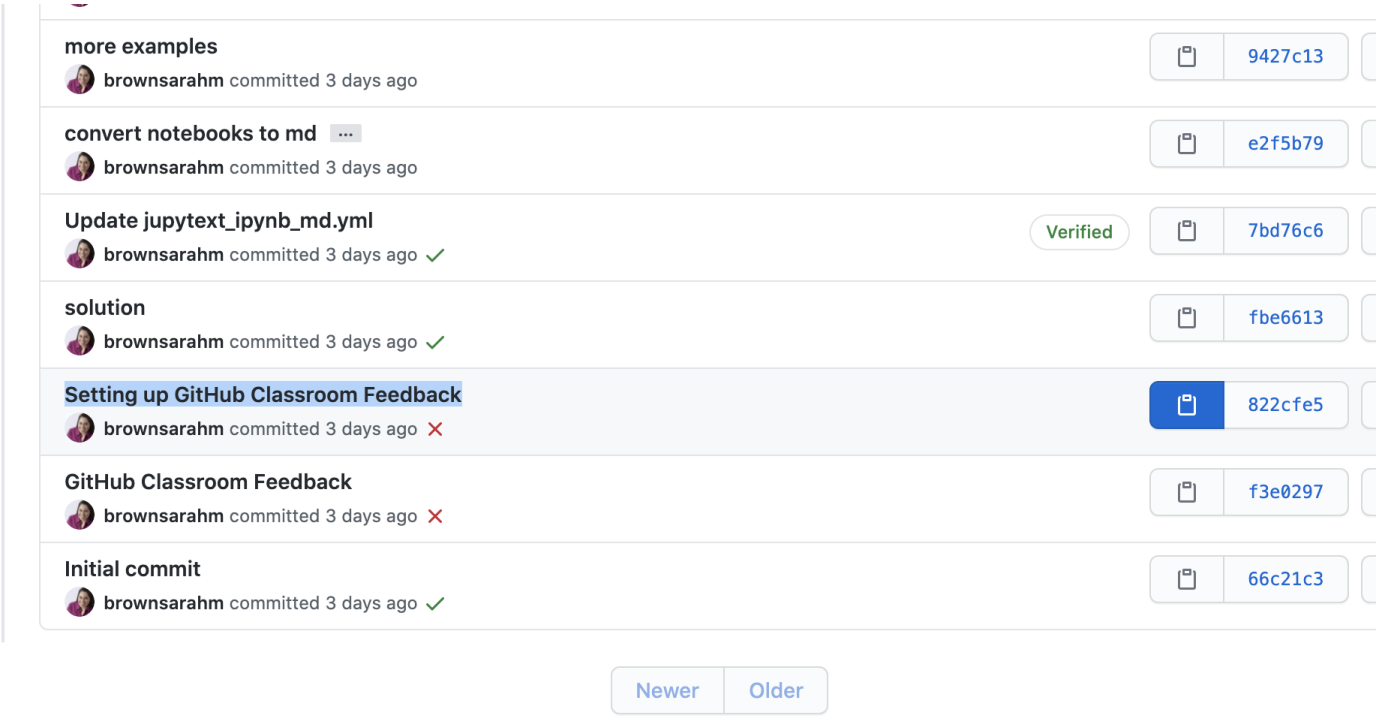
Now, you have to figure out what commit to revert to, so go back to GitHub in your browser, and swithc to the feedback branch there. Click on where it says `main` on the top right next to the branch icon and choose feedback from the list.



Now view the list of all of the commits to this branch, by clicking on the clock icon with a number of commits



On the commits page scroll down and find the commit titled “Setting up GitHub Classroom Feedback” and copy its hash, by clicking



Now, back on your terminal, type the following

```
git reset --hard
```

then paste the commit hash you copied, it will look something like the following, but your hash will be different.

```
git reset --hard 822cfe51a70d356d448bcaede5b15282838a5028
```

If it works, your terminal will say something like

```
HEAD is now at 822cfe5 Setting up GitHub Classroom Feedback
```

but the number on yours will be different.

Now your local copy of the `feedback` branch is reverted back as if you had not merged the pull request and what's left to do is to push those changes to GitHub. By default, GitHub won't let you push changes unless you have all of the changes that have been made on their side, so we have to tell Git to force GitHub to do this.

Since we're about to do something with forcing, we should first check that we're doing the right thing.

```
git status
```

and it should show something like

```
On branch feedback
Your branch is behind 'origin/feedback' by 12 commits, and can be fast-forwarded.
```

Your number of commits will probably be different but the important things to see here is that it says `On branch feedback` so that you know you're not deleting the `main` copy of your work and `Your branch is behind origin/feedback` to know that reverting worked.

Now to make GitHub match your reverted local copy.

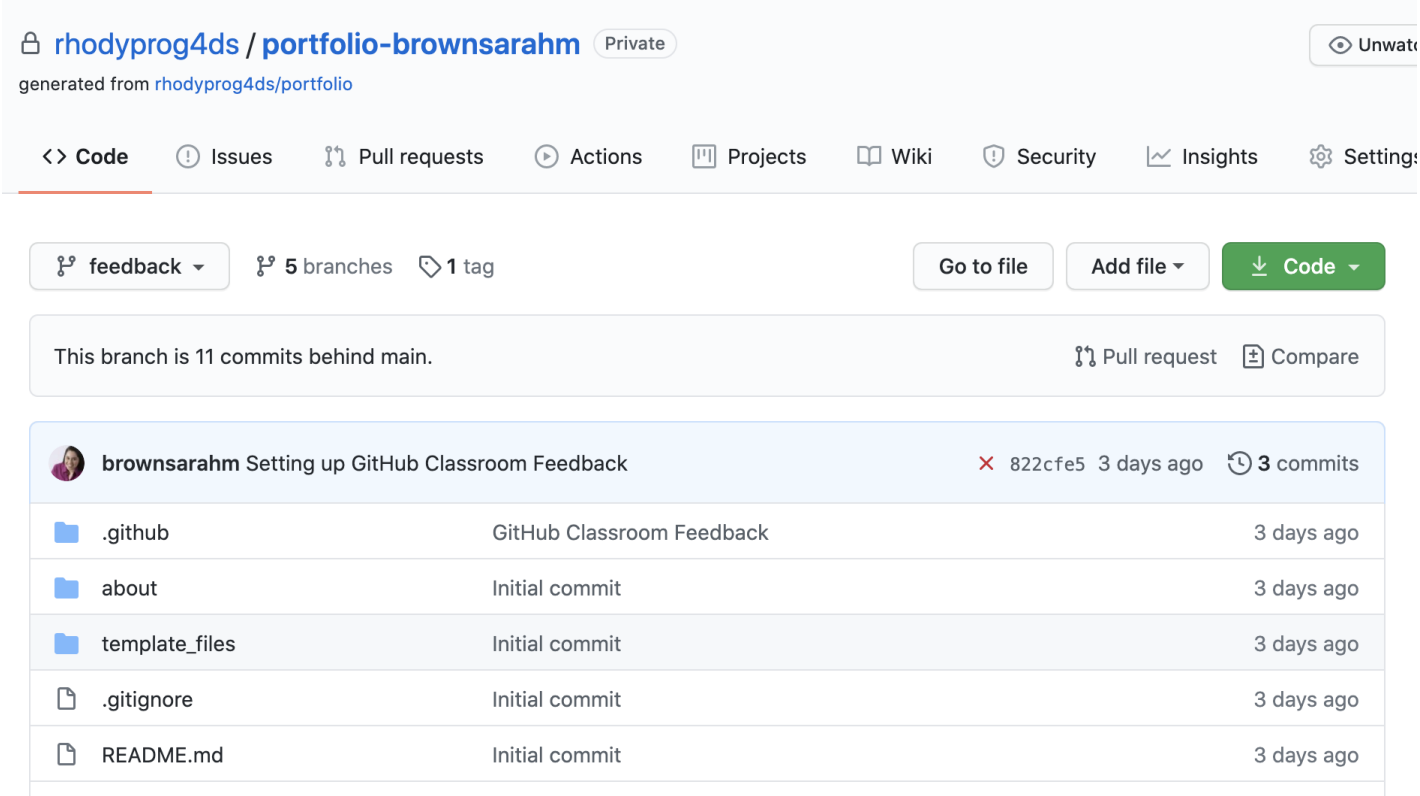
```
git push origin -f
```

and you'll get something like this to know that it worked

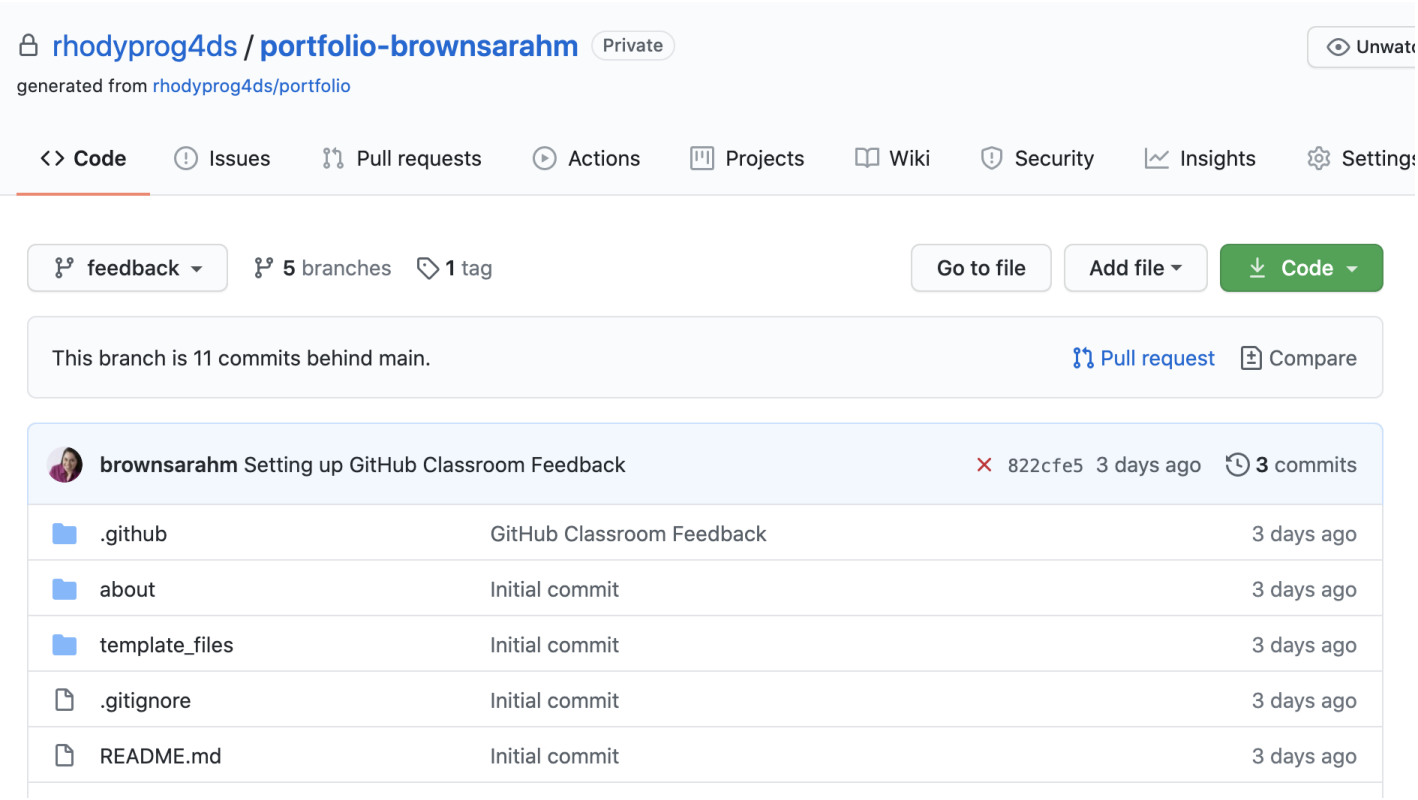
```
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/rhodyprog4ds/portfolio-brownsarahm.git
 + f301d90...822cfe5 feedback -> feedback (forced update)
```

Again, the numbers will be different and it will be your url, not mine.

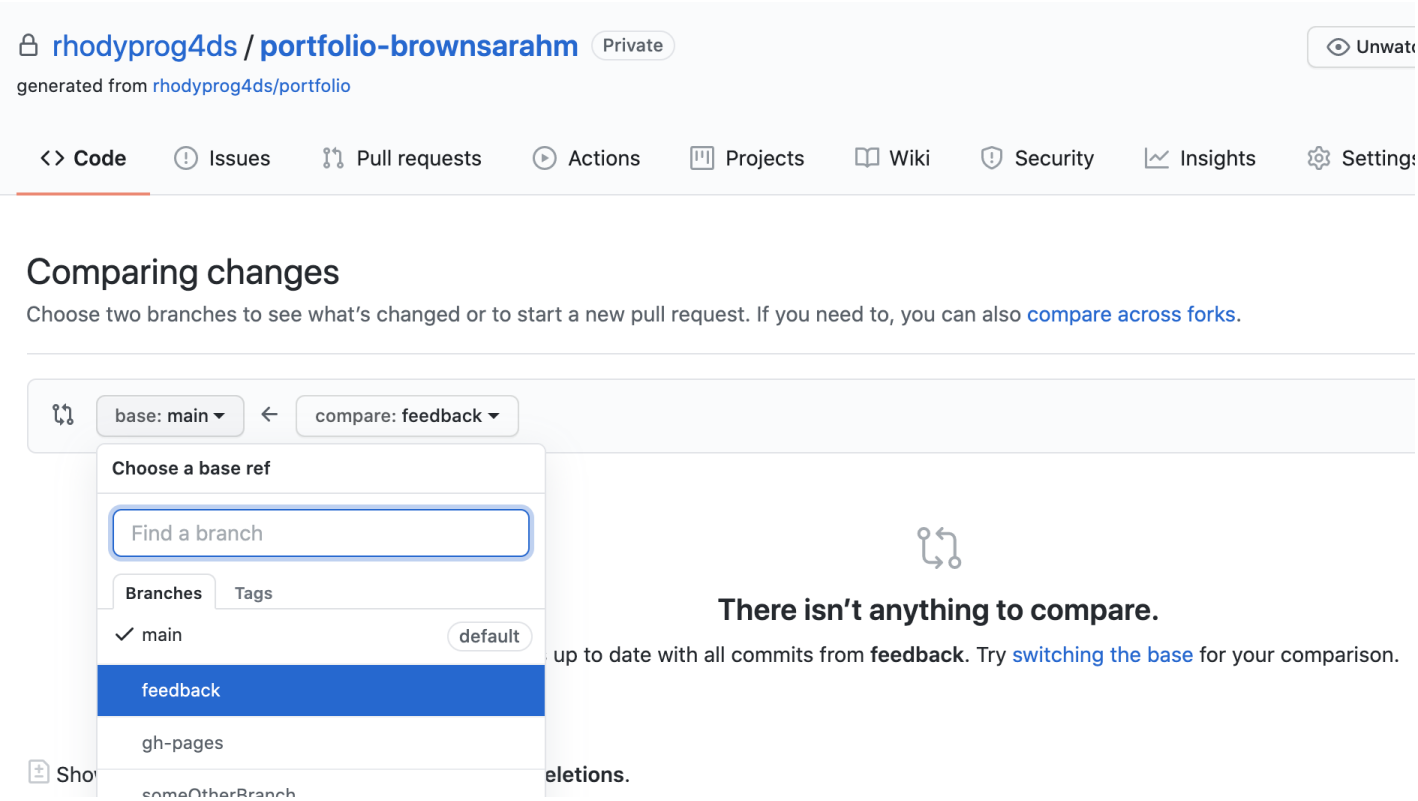
Now back on GitHub, in your browser, click on the code tab. It should look something like this now. Notice that it says, "This branch is 11 commits behind main" your number will be different but it should be 1 less than the number you had when you checked `git status`. This is because we reverted the changes you made to main (11 for me) and the 1 commit for merging main into feedback. Also the last commit (at the top, should say "Setting up GitHub Classroom Feedback").



Now, you need to recreate your Pull Request, click where it says pull request.

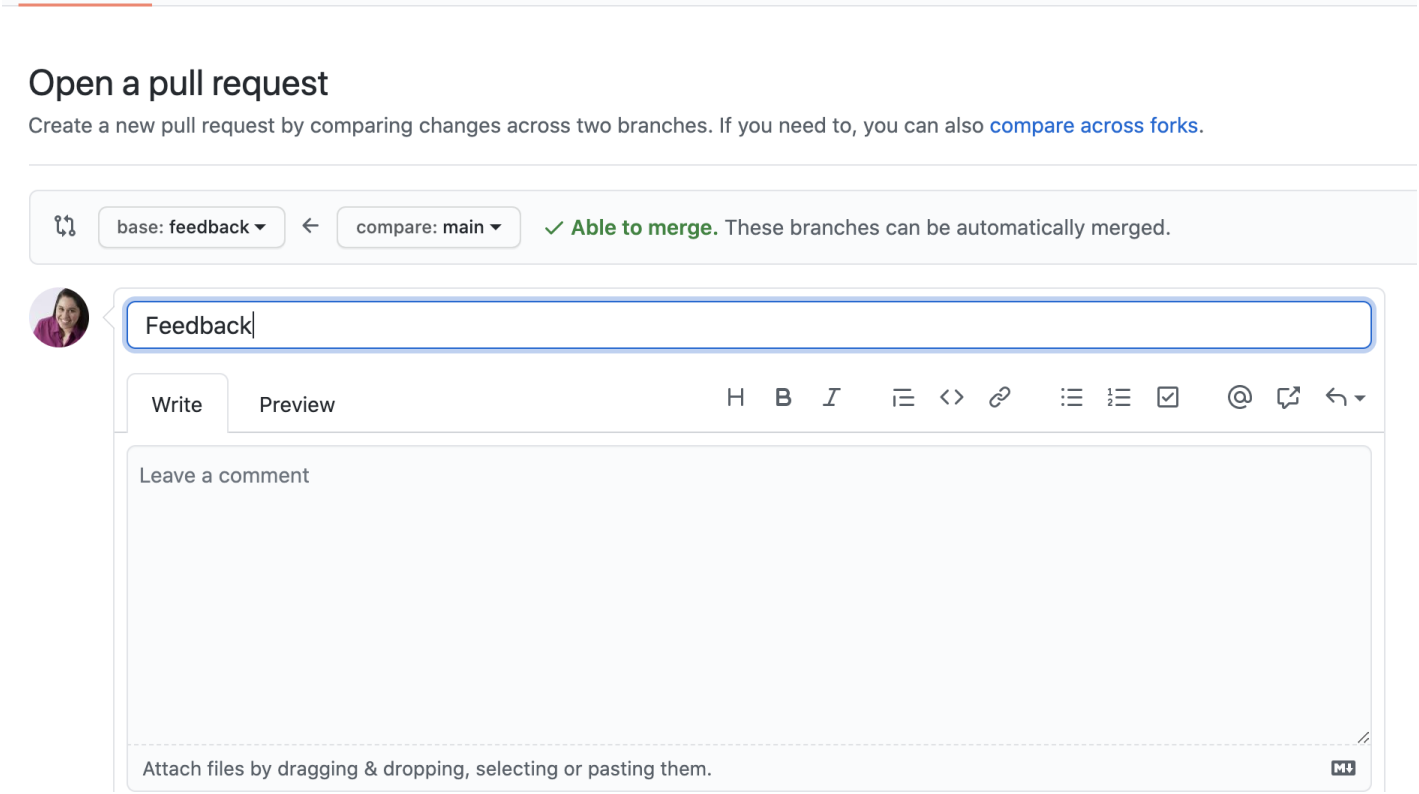


It will say there isn't anything to compare, but this is because it's trying to use `feedback` to update `main`. We want to use `main` to update `feedback` for this PR. So we have to swap them. Change base from `main` to `feedback` by clicking on it and choosing `feedback` from the list.



Then the change the compare `feedback` on the right to `main`. Once you do that the page will change to the "Open a Pull Request" interface

Skip to main content



Make the title “Feedback” put a note in the body and then click the green “Create Pull Request” button.

Now you’re done!

If you have trouble, create an issue and tag `@@rhodyprog4ds/fall22instructors` for help.

Code Errors

Key Error

If you get a key error for a pandas operation, it means that the column name as you typed it is not in the DataFrame. Check the spelling, leading or trailing whitespace can be especially troubling.

<bound method

You’re probably missing `()` on a method, so Python returned the method itself as an object instead of calling it and returning the output.

Glossary

> **Ram Token Opportunity**
>
> Contribute glossary items and links for further reading using the suggest an edit button behind the GitHub menu at the top of the page

Skip to main content

aggregate

to combine data in some way, a function that can produce a customized summary table

anonymous function

a function that's defined on the fly, typically to lighten syntax or return a function within a function. In python, they're defined with the lambda keyword.

BeautifulSoup

a python library used to assist in web scraping, it pulls data from html and xml files that can be parsed in a variety of different ways using different methods.

conditional

a logical control to do something, conditioned on something else, for example the `if`, `elif` `else`

corpus

(NLP) a set of documents for analysis

DataFrame

a data structure provided by pandas for tabular data in python.

dictionary

(data type) a mapping array that matches keys to values. (in NLP) all of the possible tokens a model knows

document

unit of text for analysis (one sample). Could be one sentence, one paragraph, or an article, depending on the goal

gh

GitHub's command line tools

git

a version control tool; it's a fully open source and always free tool, that can be hosted by anyone or used without a host, locally only.

GitHub

a hosting service for git repositories

index

(verb) to index into a data structure means to pick out specified items, for example index into a list or a index into a data frame. Indexing usually invovlees square brackets `[]` (noun) the index of a dataframe is like a column, but it can be used to refer to the rows. It's the list of names for the rows.

interpreter

the translator from human readable python code to something the computer can run. An interpreted language means you can work with python interactively

iterate

To do the same thing to each item in an iterable data structure, typically, an iterable type. Iterating is usually described as

Skip to main content

iterable

any object in python that can return its members one at a time. The most common example is a list, but there are others.

kernel

in the jupyter environment, the kernel is a language specific computational engine

lambda

they keyword used to define an anonymous function; lambda functions are defined with a compact syntax `<name> = lambda <parameters>: <body>`

PEP 8

Python Enhancement Proposal 8, the Style Guide for Python Code.

repository

a project folder with tracking information in it in the form of a .git file

suffix

additional part of the name that gets added to end of a name in a merge operation

Series

a data structure provided by pandas for single columnar data with an index. Subsetting a Dataframe or applying a function to one will often produce a Series

Split Apply Combine

a paradigm for splitting data into groups using a column, applying some function(aggregation, transformation, or filtration) to each piece and combinging in the individual pieces back together to a single table

stop words

Words that do not convey important meaning, we don't need them (like a, the, an,). Note that this is context dependent. These words are removed when transforming text to numerical representation

test accuracy

percentage of predictions that the model predict correctly, based on held-out (previously unseen) test data

Tidy Data Format

Tidy data is a database format that ensures data is easy to manipulate, model and visualize. The specific rules of Tidy Data are as follows: Each variable is a column, each row is an observation, and each observable unit is a table.

token

a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing (typically a word, but more gneeral)

TraceBack

an error message in python that traces back from the line of code that had caused the exception back through all of the functions that called other functions to reach that line. This is sometimes call tracing back through the stack

training accuracy

Skip to main content

Web Scraping

the process of extracting data from a website. In the context of this class, this is usually done using the python library beautiful soup and a html parser to retrieve specific data.

References on Python

Official Documentation

- Python
- Pandas
- Matplotlib
- Seaborn

Key Resources

- Course Text this book roughly covers things that we cover in the course, but since things change quickly, we don't rely on it too closely
- Real Python this site includes high quality tutorials
- Towards Data Science this blog has some good tutorials, but old ones are not always updated, so always check the date and don't rely too much on posts more than 2 years old.

> **Ram Token Opportunity**
>
> If you find other high quality, reliable sources that you want to share, you can earn ram tokens.

Cheatsheet

Patterns and examples of how to use common tips in class

How to use brackets

symbol	use
`[val]`	indexing item val from an object; `val` is int for iterables, or any for mapping
`[val : val2]`	slicing elemtns val to val2-1 from a listlike object
`[ item1,item2 ]`	creating a list consisting of `item1` and `item2`
`(param)`	function calls
`(item1,item2)`	defining a tuple of `item1` and `item2`
`{item1,item2}`	defining a set of `item1` and `item2`
`{key:val1, key2: val2}`	defining a dictionary where key1 indexes to val2

Axes

First build a small dataset that's just enough to display

```
data = [[1,0],[5,4],[1,4]]
df = pd.DataFrame(data = data,
  columns = ['A','B'])

df
```

	A	B
0	1	0
1	5	4
2	1	4

This data frame is originally 3 rows, 2 columns. So summing across rows will give us a Series of length 3 (one per row) and long columns will give length 2, (one per column). Setting up our toy dataset to not be a square was important so that we can use it to check which way is which.

```
df.sum(axis=0)
```

```
A    7
B    8
dtype: int64
```

```
df.sum(axis=1)
```

```
0    1
1    9
2    5
dtype: int64
```

Skip to main content

```
df.apply(sum,axis=0)
```

```
A    7
B    8
dtype: int64
```

```
df.apply(sum,axis=1)
```

```
0    1
1    9
2    5
dtype: int64
```

Indexing

```
df['A'][1]
```

```
5
```

```
df.iloc[0][1]
```

```
0
```

Data Sources

This page is a semi-curated source of datasets for use in assignments. The different sections have datasets that are good for different assignments.

Best for loading directly into a notebook

- Tidy Tuesday inside the folder for each year there is a README file with list of the datasets. These are .csv files
- Json Datasets
- National Center for Education Statistics Digest 2019 These data tables are available for download as excel and visible on the page.
- Lots of wikipedia pages have tables in them.

Cleaning Examples

- Messy Artists .csv file, that needs to be cleaned, containing data on artists
- Messy Wheels .csv file, that needs to be cleaned, containing data on various wheel attractions around the globe

Skip to main content

- Clean Wheels, .csv file, already cleaned, containing data on various wheel attractions around the globe
- Women's Rugby
- Web page metrics
- data cleaning with open refine on survey data this is a tutorial for cleaning data with another tool, but it demonstrates common problems with data well.
- data clearning for ecology this is a tutorial for cleaning data with another tool, but it demonstrates common problems with data well.
- us solar data
- NYT Data Preparation document
- Corporate Repuation Rankings

General Sources

These may require some more work

- Stackoverflow Developer Survey This data comes with readme info all packaged together in a .zip. You'll need to unzip it first.
- Google Dataset Search
- Kaggle most Kaggle datasets will require you to download and unzip them first and then you can copy them into your repo folder.
- UCI Data Repository Machine Learning focused datasets, can filter by task
- A curated list of datasets by task It includes datasets for cleaning, visualization, machine learning, and "data analysis" which would align with EDA in this course.
- Hugging Face NLP Datasets lots of text datasets

Datasets in many parts

- Makeup Shades
- Kenya Census
- Wealth and Income over time
- UN Votes
- Deforestation
- Survivor
- Billboard
- Caribou Tracking
- Video games from steam 2021 and from 2019
- BBC Rap Artists
- character psychometrics
- weather forecast accuracy

Datasets with time

- Superbowl commercials

Skip to main content

Databases

- SQLite Databases

If you have others please share by creating a pull request or issue on this repo (from the GitHub logo at the top right, `suggest edit`).

General Tips and Resources

This section is for materials that are not specific to this course, but are likely useful. They are not generally required readings or installs, but are options or advice I provide frequently.

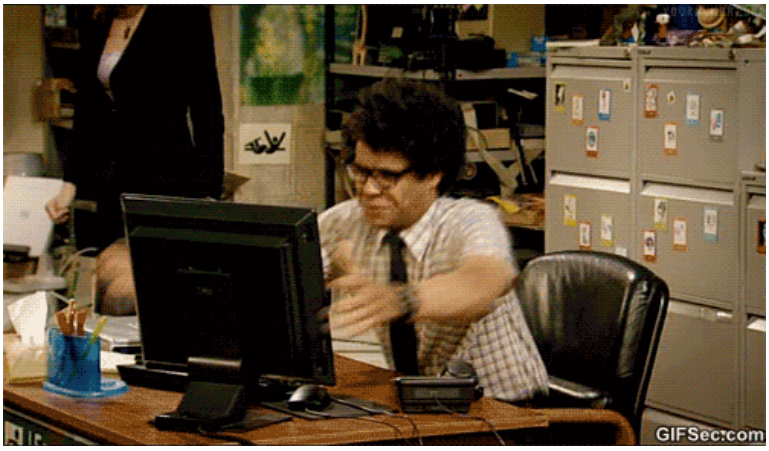
on email

- how to e-mail professors

How to Study in this class

This is a programming intensive course and it's about data science. This course is designed to help you learn how to program for data science and in the process build general skills in both programming and using data to understand the world. Learning two things at once is more complex. In this page, I break down how I expect learning to work for this class.

Remember the goal is to avoid this:



Why this way?

Learning to program requires iterative practice. It does not require memorizing all of the specific commands, but instead learning the basic patterns.
Using reference materials frequently is a built in part of programming, most languages have built in help as a part of the language for this reason. This course is designed to have you not only learn the material, but also to build skill in learning to program. Following these guidelines will help you build habits to not only be successful in this class, but also in future programming.

> **Where are your help tools?**
>
> In Python and Jupyter notebooks, what help tools do you have?

A new boo
programmi
Brain As of
by clicking
contents s

Skip to main content

Learning in class

> **Important**
>
> My goal is to use class time so that you can be successful with *minimal frustration* while working outside of class time.

Programming requires both practical skills and abstract concepts. During class time, we will cover the practical aspects and introduce the basic concepts. You will get to see the basic practical details and real examples of debugging during class sessions. Learning to debug something you've never encountered before and setting up your programming envrionment, for example, are *high frustration* activities, when you're learning, because you don't know what you don't know. On the other hand, diving deeper into options and more complex applications of what you have already seen in class, while challenging, is something I'm confident that you can all be successful at with minimal frustration once you've seen basic ideas in class. My goal is that you can repeat the patterns and processes we use in class outside of class to complete assignments, while acknowledging that you will definitely have to look things up and read documentation outside of class.

Each class will open with some time to review what was covered in the last session before adding new material.

To get the most out of class sessions, you should have a laptop with you. During class you should be following along with Dr. Brown, typing and running the same code. You'll answer questions on Prismia chat, when you do so, you should try running necessary code to answer those questions. If you encounter errors, share them via prismia chat so that we can see and help you.

After class

After class, you should practice with the concepts introduced.
This means reviewing the notes: both yours from class and the annotated notes posted to the course website.
When you review the notes, you should be adding comments on tricky aspects of the code and narrative text between code blocks in markdown cells. While you review your notes and the annotated course notes, you should also read the documentation for new modules, libraries, or functions introduced that day.

In the annotated notes, there will often be extra questions or ideas on how to extend and practice the concepts. Try these out.

If you find anything hard to understand or unclear, write it down to bring to class the next day.

Assignments

In assignments, you will be asked to practice with specific concepts at an intermediate level. Assignments will apply the concepts from class with minimal extensions. You will probably need to use help funcitons and read documentation to complete assignments, but mostly to look up things you saw in class and make minor variations. Most of what you need for assignments will be in the class notes, which is another reason to read them after class.

Portfolios

In portfolios, your goal is to extend and apply the concepts taught in class and practiced in assignments to solve more realistic problms. You may also reflect on your learning in order to demonstrate deep understanding. These will require significant reading beyond what we cover in class.

Skip to main content

Getting Help with Programming

Asking Questions

One of my favorite resources that describes how to ask good questions is this blog post by Julia Evans, a developer who writes comics about the things she learns in the course of her work and publisher of wizard zines.

Describing what you have so far

Stackoverflow is a common place for programmers to post and answer questions.
As such, they have written a good guide on creating a minimal, reproducible example.

Creating a minimal reproducible example may even help you debug your own code, but if it does not, it will definitely make it easier for another person to understand what you have, what your goal is, and what's working.

Understanding Errors

Error messages from the compiler are not always straight forward.

The TraceBack can be a really long list of errors that seem like they are not even from your code. It will trace back to all of the places that the error occurred. It is often about how you called the functions from a library, but the compiler cannot tell that.

Skip to main content

One thing to try, is friendly traceback a python package that is designed to make that error message text more clear and help you figure out what to do next.

Ram Token Opportunity

If you try out friendly traceback and find it helpful, add a testimonial here. using

```
```{epigraph}
```
```

Terminals and Environments

Why all this work?

Managing environments is **one of the hardest parts of programming** so, as instructors, we often design our courses around not having to do it. In this class, however, I'm choosing to take the risk and help you all through beginning to manage your own environments.

These issues will be the most painful in the course, I promise.

I think it's worth this type of pain though, because all fo the code you ever run must run in *some* sort of environment. By giving you control, I'm hoping to increase your indepence as a programmer. This also means responsibility and some messy debugging, but I think this is a good tradeoff. This is an upper level (300+) level course, so increasing some complexity is expected and I want as much as possible to keep you close to realisitc programming environments; so that what you see in this course is **directly, and immediately,** applicable in real world contexts. You should be able to pick up data science side projects or an internship with ease after this course.

I *know* some of these things will be frustrating at times, but I want you to feel supported in that and know that your grade will not be blocked by you having environment issues, as long as you ask for help in a timely matter.

Not

We k
teach
so in
new c
that w
under
this s
follow
and th

Windows

Windows has a sort of multiverse of terminal environments.

The least setup required involves using anaconda prompt and `conda` to manage you python environment and GitBash to work with git (and it can also do other bash related things).

Instead of managing two terminals, you may configure your path in GitBash to make Anaconda work

If, for exam
have neve
and you're
will be hurt
at that poir

MacOS

MacOS has one terminal app, but it can run different shells.

On MacOS You may want to switch to bash (using the `bash` command or make it your default and update bash.

Skip to main content

Getting Organized for class

The only **required** things are in the Tools section of the syllabus, but this organizational structure will help keep you on top of what is going on.

Your username will be appended to the end of of the repository name for each of your assignments in class.

File structure

I recommend the following organization structure for the course:

```
CSC310
  |- notes
  |- portfolio-username
  |- 02-accessing-data-username
  |- ...
```

This is one top level folder will all materials in it. A folder inside that for in class notes, and one folder per repository.

Please **do not** include all of your notes or your other assignments all inside your portflio, it will make it harder to grade.

Finding repositories on github

Each assignment repository will be created on GitHub with the `rhodyprog4ds` organization as the owner, not your personal acount. Since your account is not the owner, they do not show on your profile.

Your assignment repositories are all private during the semester. At the end, you may take ownership of your portfolio[^pttrans] if you would like.

If you go to the main page of the organization you can search by your username (or the first few characters of it) and see only your repositories.

> ⚠ **Warning**
>
> Don't try to work on a repository that does not end in your username; those are the template repositories for the course and you don't have edit permission on them.