Contents

About this Book

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

This class meets MWF 3-3:50pm in Chafee Social Sci Center 235.

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

Course Calendar on BrightSpace

Tip

subscribe to that calendar in your favorite calendar application

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

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.

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. Short questions will be in the margin note

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.

> **Ram Token Opportunity**
>
> Chances to earn ram tokens are highlighted this way.

Basic Facts

About this course

Data science exists at the intersection of computer science, statistics, and machine learning. 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.

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 substantial 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 this syllabus

This syllabus is a *living* document and accessible from BrightSpace, as a pdf for download directly online at rhodyprog4ds.github.io/BrownFall21/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 You can view the date of changes and exactly what changes were made on the Github commit history page.

Creating an issue 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 M Brown Office hours: TBA via zoom, link on BrightSpace

Dr. Sarah M Brown is a second year 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. You can learn more about me at my website or my research on my lab site.

You can call me Professor Brown or Dr. Brown, I use she/her pronouns.

The best way to contact me is e-mail or an issue on an assignment repo. For more details, see the Communication Section

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 all other materials. Any links that are for private discussion among those enrolled in the course will be available only from our course Brightspace site.

This is also where your grades will appear and how I will post announcements.

For announcements, you can customize how you receive them.

> **Important**
>
> TL;DR [1]
>
> - check Brightspace
> - Log in to Prismia Chat
> - Make a GitHub Account
> - Install Python
> - Install Git

> **Note**
>
> Seeing the BrightSpace site requires loging in with your URI SSO and being enrolled in the course

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 Manual

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/BrownFall21/. Links to the course reference text and code documentation will also be included here in the assignments and class notes.

GitHub Classroom

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 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)
- Git
- 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.

> **Note**
>
> all Git instructions will be given as instructions for the command line interface and GitHub specific instructions via the web interface. You may choose to use GitHub desktop or built in IDE tools, but the instructional team may not be able to help.

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`

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.

Video install instructions for Anaconda:

- 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.

A tip from Dr. Brown

I use atom, but I decided to use it by downloading both Atom and Sublime and trying different things in each for a week. I liked Atom better after that and I've stuck with it since. I used Atom to write all of the content in this syllabus. VScode will also work, if needed

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 only, Fall 2021 is in person)

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 MWF 3-3:50pm in Chafee Social Sci Center 235. 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.

Programming assignments that will be due each week Tuesday by 11:59pm.

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, Naive Bayes, classification performance metrics	[classification, evaluate]
7	decision trees, cross validation	[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]

Note

On the Course Calendar on BrightSpace page you can get a feed link to add to the calendar of your choice by clicking on the subscribe (star) button on the top right of the page. Class is for 1 hour there because of Brightspace/zoom integration limitations, but that calendar includes the zoom link.

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
classification	Apply classification	identify and describe what classification is, apply pre-fit classification models	fit 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	can fit linear regression models	can fit and explain regrularized or nonlinear regression
clustering	Clustering	describe what clustering is	apply basic clustering	apply multiple clustering techniques, and interpret results
evaluate	Evaluate model performance	Explain basic performance metrics for different data science tasks	Apply basic model evaluation metrics to a held out test set	Evaluate a model with multiple metrics and cross validation
optimize	Optimize model parameters	Identify when model parameters need to be optimized	Manually optimize basic model parameters such as model order	Select optimal parameters based of mutiple quanttiateve criteria and automate parameter tuning
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

keyword	skill	Level 1	Level 2	Level 3
unstructured	model unstructured data	Identify options for representing text data and use them once data is tranformed	Apply at least one representation to transform unstructured data for model fitting or summarizing	apply multiple representations and compare and contrast them for different end results
workflow	use industry standard data science tools and workflows to solve data science problems	Solve well strucutred problems with a single tool pipeline	Solve semi-strucutred, completely specified problems, apply common structure to learn new features of standard tools	Scope, choose an appropriate tool pipeline and solve data science problems, 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	1	1	0	0	0	0	0	0	0	0	0	4
process	1	1	0	0	0	0	0	0	0	1	1	0	0	4
access	0	1	1	1	0	0	0	0	0	0	0	0	0	3
construct	0	0	0	0	1	1	0	0	0	0	0	0	0	2
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
classification	0	0	0	0	0	1	1	0	0	1	0	0	0	3
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
evaluate	0	0	0	0	0	0	0	0	0	1	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
unstructured	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

> ⚠️ **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	0
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	0
construct	merge data that is not automatically aligned	1	1	0	0
summarize	Compute and interpret various summary statistics of subsets of data	1	1	0	0
visualize	generate complex plots with pandas and plotting libraries and customize with matplotlib or additional parameters	1	1	0	0
prepare	apply data reshaping, cleaning, and filtering manipulations reliably and correctly by assessing data as received	1	1	0	0
classification	fit and apply classification models and select appropriate classification models for different contexts	0	1	1	0
regression	can fit and explain regrularized or nonlinear regression	0	1	1	0
clustering	apply multiple clustering techniques, and interpret results	0	1	1	0
evaluate	Evaluate a model with multiple metrics and cross validation	0	1	1	0
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
unstructured	apply multiple representations and compare and contrast them for different end results	0	0	1	1
workflow	Scope, choose an appropriate tool pipeline and solve data science problems, describe strengths and weakensses of common tools	0	0	1	1

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 not require you to get anything on the first try or to explore topics on your own. I expect most students to reach this level.
- 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.

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.

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.

> **Warning**
>
> If you will skip an assignment, please accept the GitHub assignment and then close the Feedback pull request with a comment. This way we can make sure that you have support you need.

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 achieved mastery of all of the skills by the 3rd portfolio check, you do not need to submit the fourth one.

Portfolio prompts will be given throughout the class, some will be strucutred questions, others may be questions that arise in class, for which there is not time to answer.

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

On Brightspace there are 45 Grade items that you will get a 0 or a 1 grade for. These will be revealed, so that you can view them as you have an opportunity to demonstrate each one. The table below shows the minimum number of skills at each level to earn each letter grade.

	Level 3	Level 2	Level 1
letter grade			
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.

> **Note**
>
> In this example, you will have also achieved level 1 on all of the skills, because it is a prerequisite to level 2.

The letter grade can be computed as follows

```
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
```

For example you can run the code like this in a cell to see the output

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

```
'A'
```

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

```
'C-'
```

Or use `assert` to test it formally

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

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

```
assert compute_grade(15,15,11) == 'A-'
```

Late work

Late assignments will not be graded. Every skill will be assessed through more than one assignment, so missing assignments occasionally not necessarily hurt 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 not be able to earn the level 3 achievement in that skill. If you submit work that is not complete, however, it will be assessed and receive feedback. Submitting pseudocode or code with errors and comments about what you have tried could earn a level 1 achievement. Additionally, most 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.

> **Note**
>
> You may visit office hours to discuss assignments that you did not complete on time to get feedback and check your own understanding, but they will not count toward skill demonstration.

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.

In this issue, include:

1. A new deadline proposal
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 assignment on time.

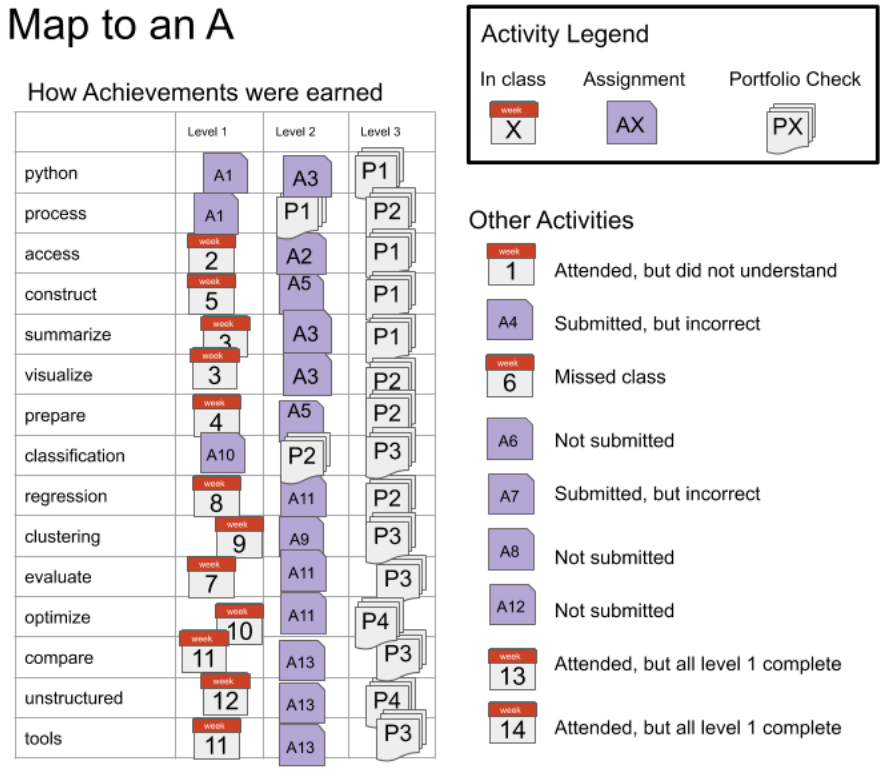
This request should be no more than 7 sentences.

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.

Grading Examples

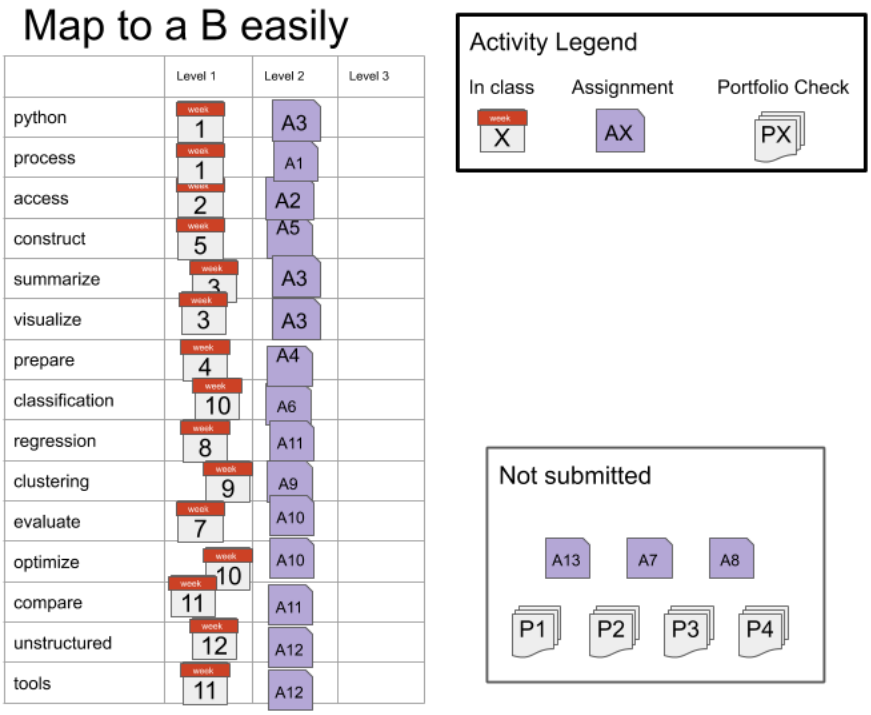
If you always attend and get everything correct, you will earn and A and you won't need to submit the 4th portfolio check or assignment 13.

Getting an A Without Perfection



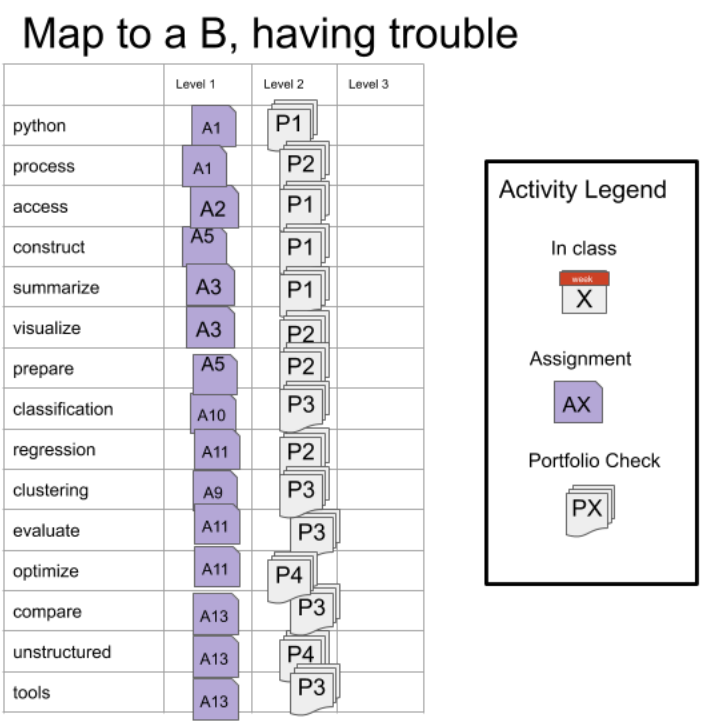
In this example the student made several mistakes, but still earned an A. This is the advantage to this grading scheme. For the `python`, `process`, and `classification` skills, the level 1 achievements were earned on assignments, not in class. For the `process` and `classification` skills, the level 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



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



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.

Ram Tokens

Ram Tokens in this course will be used as a currency for extra effort. You can earn Ram Tokens by doing work that supports your learning or class activities, but do not directly demonstrate achievements. You can spend Ram Tokens to get extra grading. This will be mostly applicable to Portfolio Checks. In Checks 3 & 4, some achievements will not be eligible for grading as per the table. However, you can exchange Ram Tokens to make more achievements eligible for assessment. This system rewards you for putting in consistent effort, even if it takes you many tries to understand a concept.

To accumulate Ram Tokens, you submit a 'Deposit' to the Ram Token Bank: http://drsmb.co/ramtoken with a link to what you did to earn a token. To apply Ram tokens for extra grading, submit the same form, with a link to the assignment and add the Ramtoken label to the Feedback PR.

Support

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. Appointments and locations will be visible in the TutorTrac system on September 14th, 2020. 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.

Policies

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.

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
- Facilitating or aiding another's academic dishonesty
- Submitting the same paper for more than one course without prior approval from the instructors

URI COVID-19 Statement

The University is committed to delivering its educational mission while protecting the health and safety of our community. While the university has worked to create a healthy learning environment for all, it is up to all of us to ensure our campus stays that way.

As members of the URI community, students are required to comply with standards of conduct and take precautions to keep themselves and others safe. Visit web.uri.edu/coronavirus/ for the latest information about the URI COVID-19 response.

- Universal indoor masking is required by all community members, on all campuses, regardless of vaccination status. If the universal mask mandate is discontinued during the semester, students who have an approved exemption and are not fully vaccinated will need to continue to wear a mask indoors and maintain physical distance.
- Students who are experiencing symptoms of illness should not come to class. Please stay in your home/room and notify URI Health Services via phone at 401-874-2246.
- If you are already on campus and start to feel ill, go home/back to your room and self-isolate. Notify URI Health Services via phone immediately at 401-874-2246.

If you are unable to attend class, please notify me at brownsarahm@uri.edu. We will work together to ensure that course instruction and work is completed for the semester.

Course Communications

Help Hours

Day	Time	Location	Host
Monday	1:00:00 PM-2:30	inperson roomtbd	Chamundi
Wednesday	4:00:00 PM	inperson roomtbd	Chamundi
Wedneday	2:00:00 PM-3	inperson roomtbd	Chamundi
Wednesday	7:00:00 PM-8:30	gather.town	Sarah
Friday	5:00:00 PM-6:30pm	gather.town	Chamundi
By appointment	TBD	in person Tyler 134	Sarah

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	message board	for discussion with peers
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	github	pull request on website	remember to request ram tokens if applicable
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.

By Platform

Use e-mail for

usage	area	note
matters that don't fit into another category	to brownsarahm@uri.edu	remember to include `[CSC310]` or `[DSP310]` (note `verbatim` no space)

Use github for

usage	area	note
private questions to your assignment	issue on assignment repo	eg bugs in your code"
for general questions that can help others	issue on course website	eg what the instructions of an assignment mean or questions about the syllabus
to share resources	pull request on website	remember to request ram tokens if applicable

Use prismia for

usage	area	note
in class	chat	outside of class time this is not monitored closely
any time	message board	for discussion with peers
any time	download transcript	use after class to get preliminary notes eg if you miss a class

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.

> **Note**
>
> Whether you use CSC or DSP does not matter.

1. Welcome to Programming to Data Science

Today's goals:

1. Operate tools for in-class participation
2. Understand what Data Science is, in broad terms
3. Understand the syllabus (grading, topics covered, schedule, etc)
4. Understand how to learn in this course

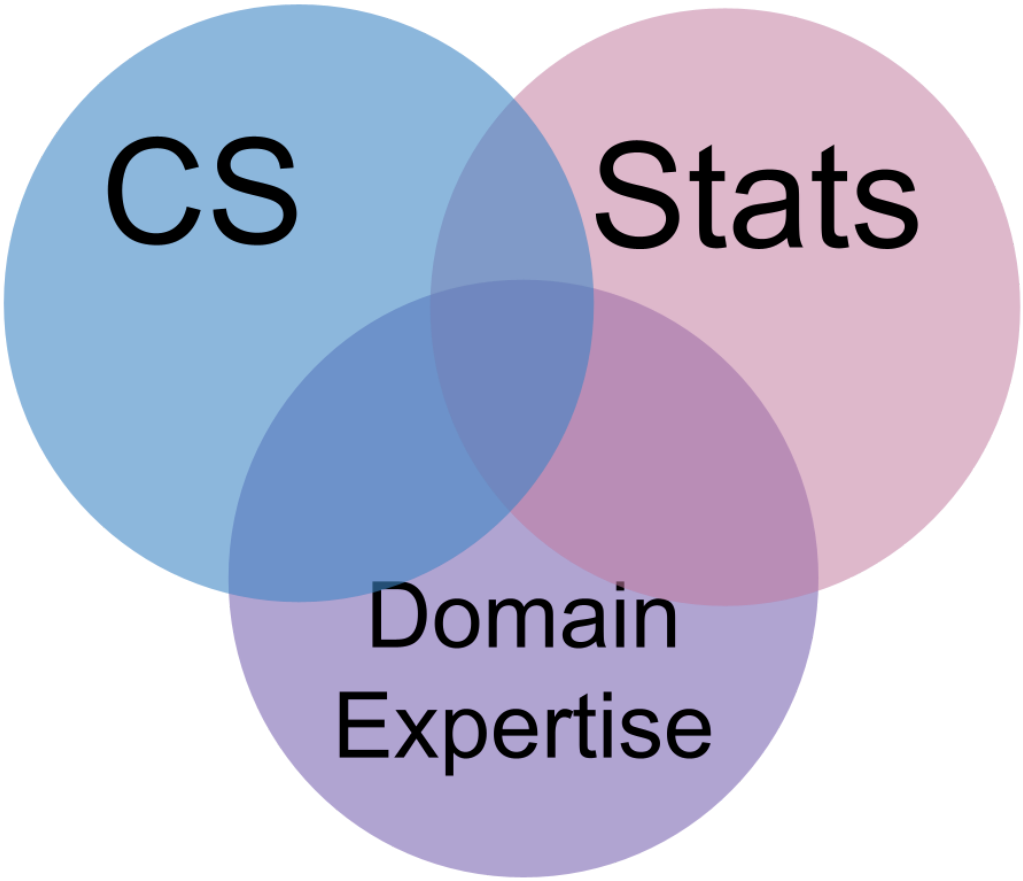
1.1. Prismia Chat

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. What is Data Science

In general:

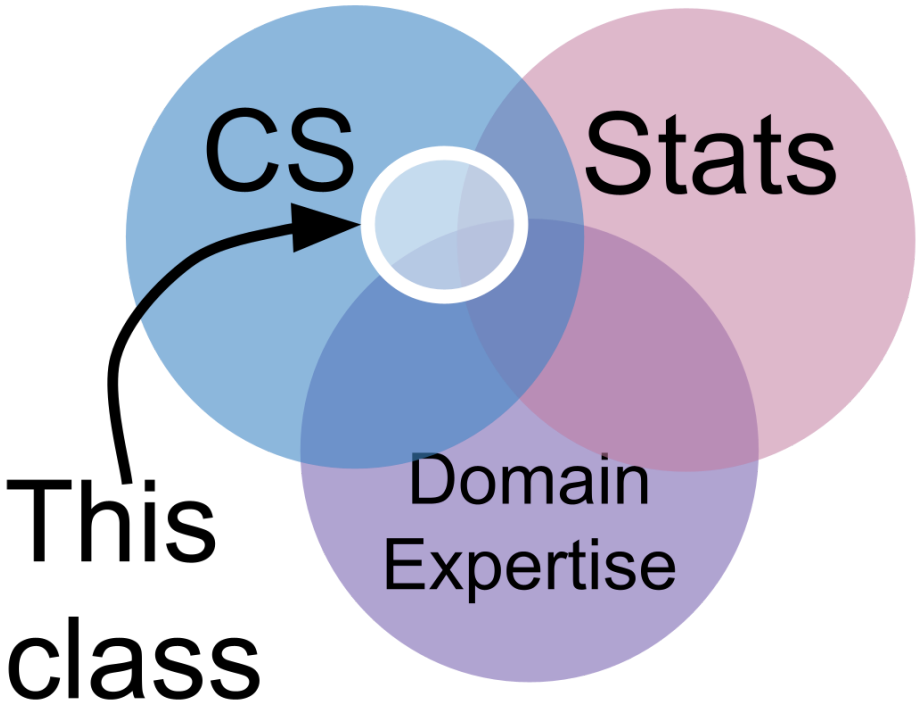


statistics is the type of math we use to make sense of data. Formally, a statistic is just a function of data.

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 worked right. A statistic must be interpreted in context; the relevant context determines what they mean and which are valid. The context will say whether automating something is safe or not, it can help us tell whether our code actually worked right or not.

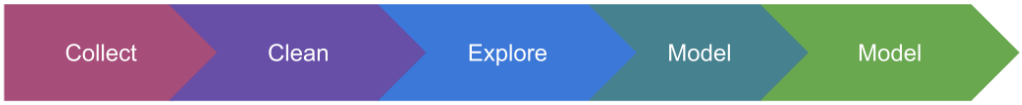
For this class



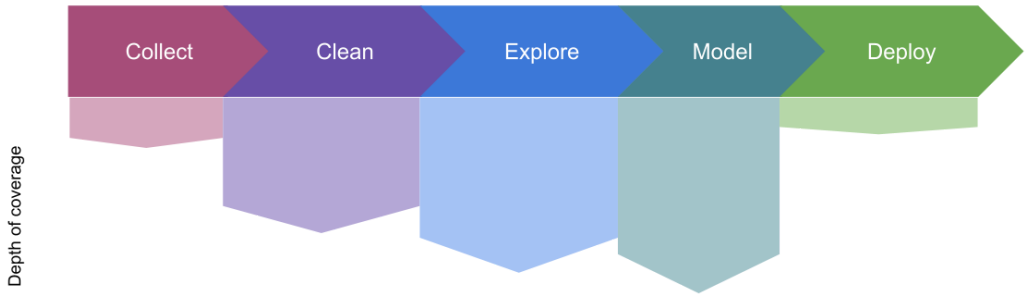
We'll focus on the programming as our main means of studying data science, but we will use bits of the other parts. In particular, you're encouraged to choose datasets that you have domain expertise about, or that you want to learn about.

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

1.2.1. How does data science happen?

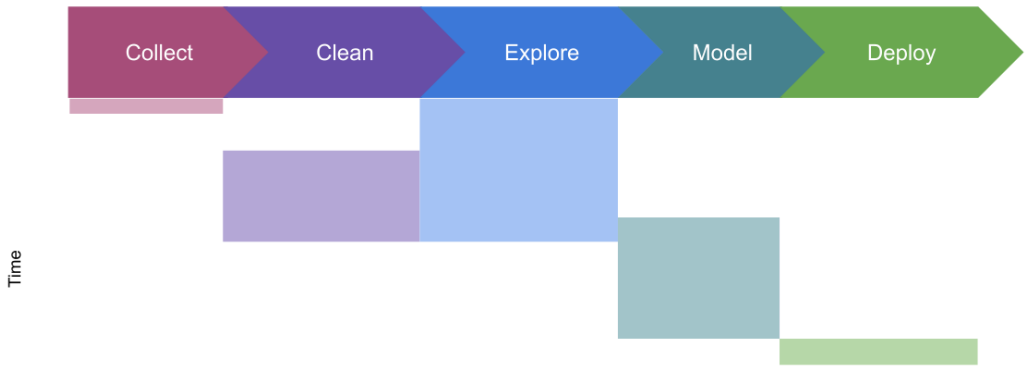


1.2.2. how we'll cover it, in depth



- *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

1.2.2.1. 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 cleaneed the data.

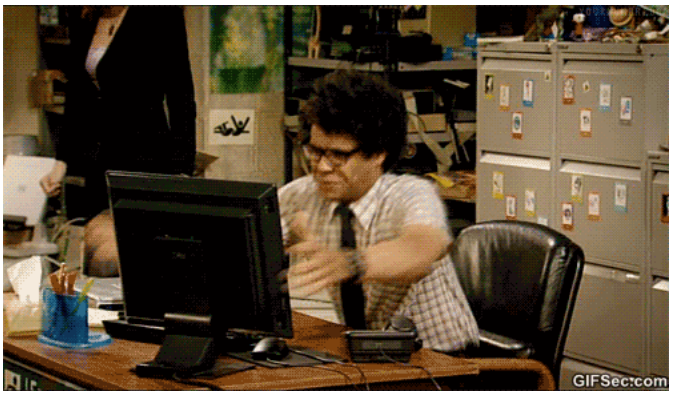
1.3. How this class will work

- today is an exception
- in general we'll be live coding

Let's look at the [syllabus]

Read carefully to make sure you understand the grading; it's not typical points and an average.

Class is designed to avoid this:

1.4. 

1.5. Learning Cycle

Read more about how I'm designing this course to help you learn on the how to learn page.

1.6. Check your understaning of the syllabus

It's easy when reading something long to lose track of it. Your eyes can go over each word, without actually retaining the information, but it's important to understand the syllabus for the course.

You can find the answers to the following questions on the syllabus. If you've already read it, try answering them to check your understanding. If you haven't read it yet, use these to guide you to get familiar with finding key facts about the course on the syllabus.

1. What do you need to bring to class each day?
2. What is the basis of grading for this course?
3. How do you reference the course text?
4. What is the penalty for missing an assignment?

More information about the course is available throughout the site, the next few questions will help you self-check that you've found the important things. Remember, the goal is not necessarily to memorize all of this, but to be able to find it.

1. When & what are you expected to read for this class?

- [ ] read the text book before class
- [ ] review notes & documentation after class
- [ ] preview the notes & documentation before class
- [ ] read documentation and text book after class

1. Your assignment says to find a dataset that has variables of a specific type, which website can you use?
2. Your assignment says to find a dataset of any type about something you're interested in, which resource would you use?

2. Jupyter Notebook Tour & Python Review

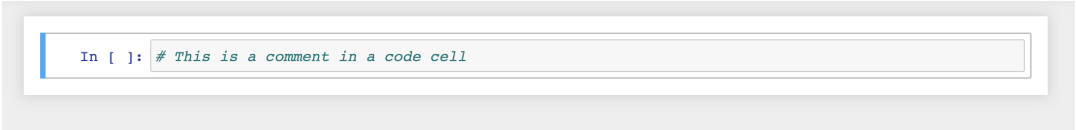
2.1. A jupyter notebook tour

Launch a `jupyter notebook`:

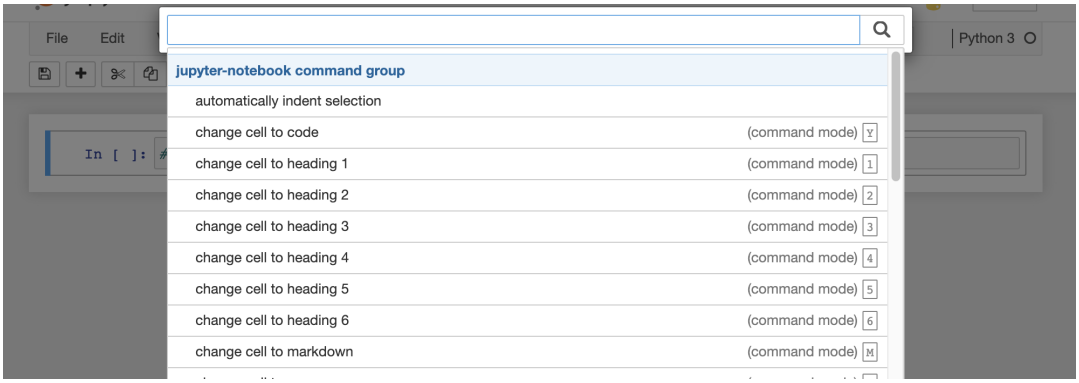
- on Windows, use anaconda terminal
- on Mac/Linux, use terminal

```
cd path/to/where/you/save/notes
jupyter notebook
```

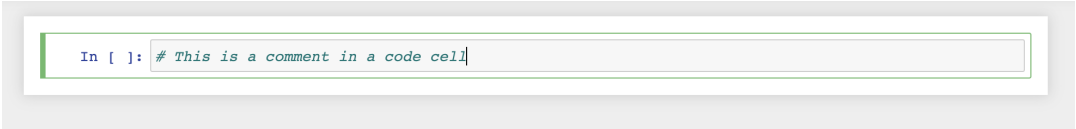
A Jupyter notebook has two modes. When you first open, it is in command mode. The border is blue in command mode.



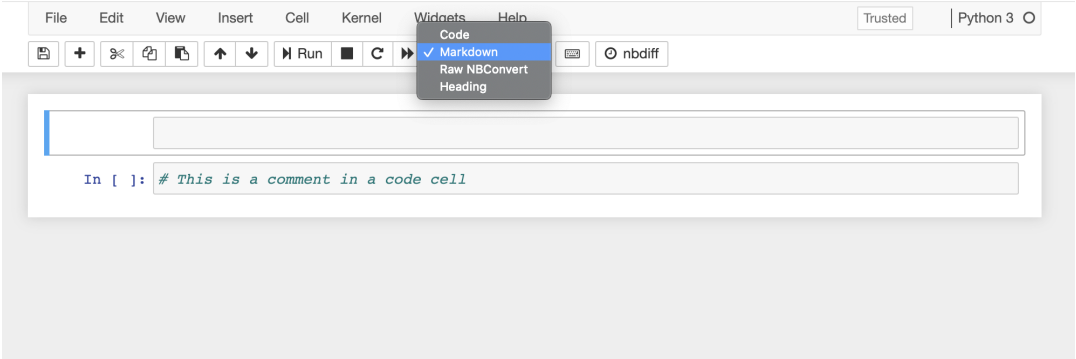
When you press a key in command mode it works like a shortcut. For example `p` shows the command search menu.



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. The border is green in 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
3. about renumbering them
4. if we add a step in the middle later

2.1.1. Notebook Reminders

Blue border is command mode, green border is edit mode

use Escape to get to command mode

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
- p: command menu

use enter/return to get to edit mode

In code cells, we can use a python interpreter, for example as a calculator.

```
4+6
```

```
10
```

It prints out the last line of code that it ran, even though it executes all of them

```
name = 'sarah'
4+5
name *3
```

```
'sarahsarahsarah'
```

> **Note**
>
> For a little more python review, see my [2020 CSC310 notes]() this is just enough for this assignment.

2.2. Just enough Git for Assignment 1

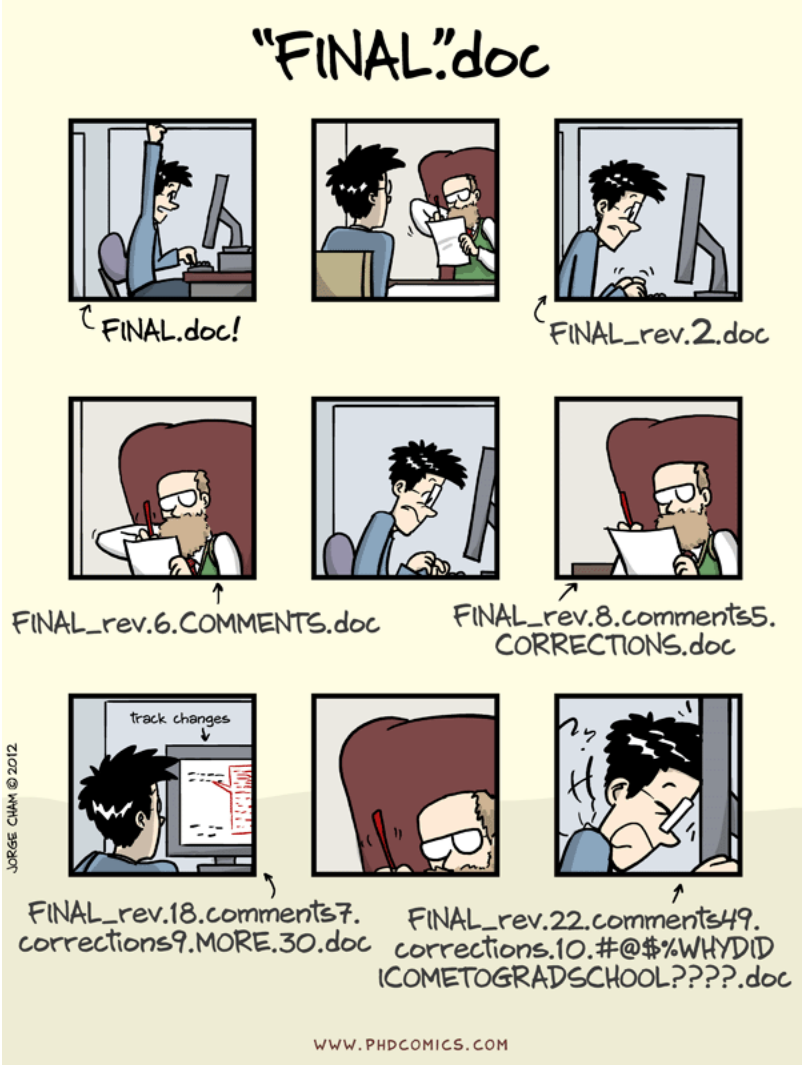
2.2.1. Assignment 1:

Goals for this assignment

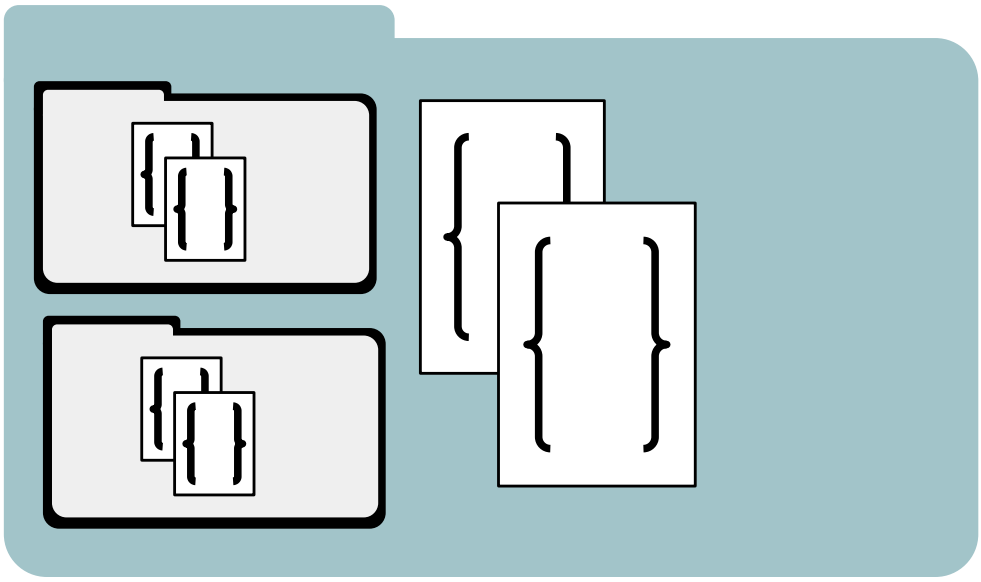
- setup your portfolio
- check that you understand the grading
- review Python basics
- practice with git and GitHub

2.2.2. Why Version control

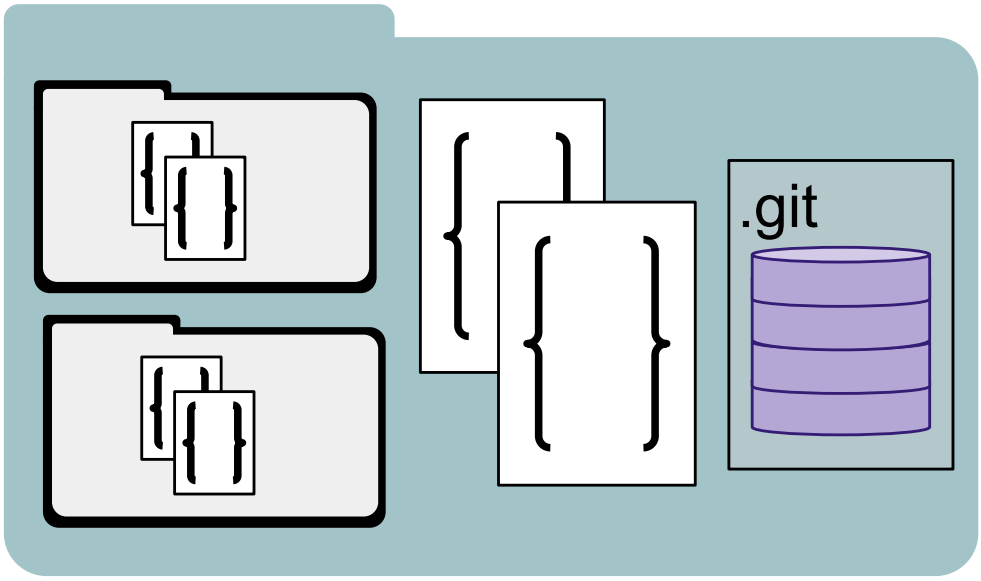
We often want to keep track of the different versions in case we want to go back, but this can be painful:



We typically organize projects in folder

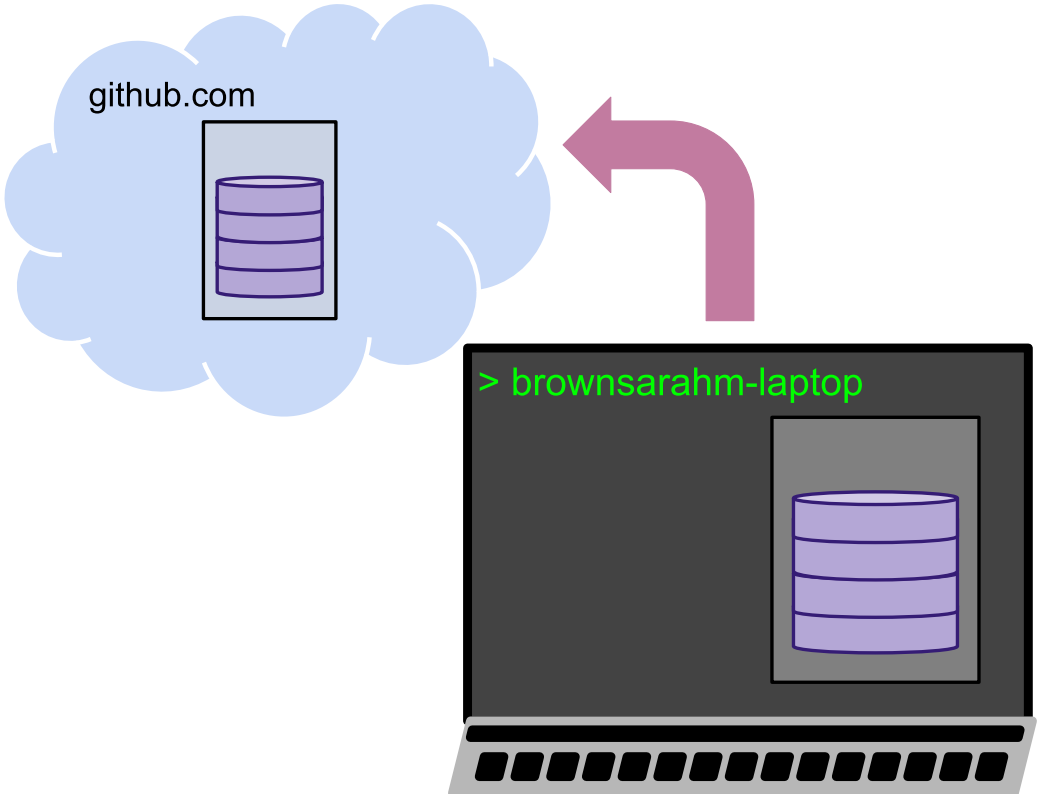


A repository is a folder with a hidden directory named `.git`

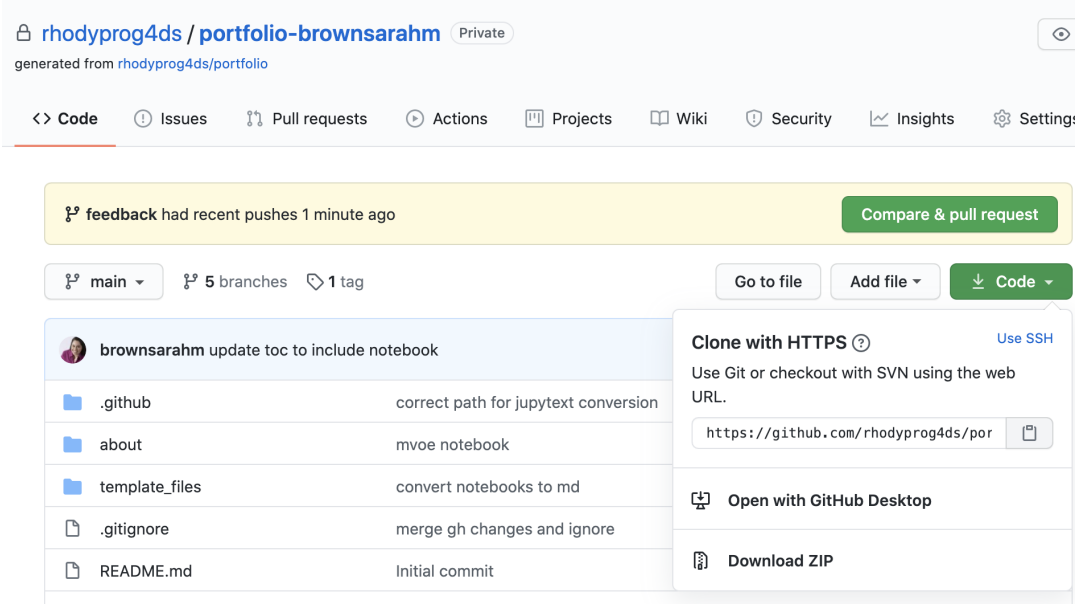


The `git` applicaiton manages that hidden directory, we don't write to it directly, which is why we keep it hidden.

Git is a distributed system, you have a local version and a remote version.



Once a repository exists on GitHub, we get a local copy by cloning it after we get its address from the GitHub interface, by clicking on the green code butto that is below the menu area to the right. It's at the top right corner of the list of files in the repository.



For this part, use GitBash on windows or terminal otherwise: If you set up a Personal Access Token you can use the https version

After `cd/to/where/you/want/your/repo/locally`:

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

If you set up ssh keys you use that instead

```
git clone git@github.com:rhodyprog4ds/portfolio-example.git
```

Once it's cloned, then you can navigate into the new folder:

```
cd portfolio-example
```

Then you can change files, for example adding to the intro.

Some common actions in Git, you'll want.

Check on the status of your repository:

```
git status
```

Add files to the staging area:

```
git add filename
```

Add all changes to the staging area:

```
git add .
```

Commit your changes to the repository:

```
git commit -m 'a message that will help your future self know what this part is'
```

Push your changes to GitHub

```
git push
```

Pull changes from GitHub

Note

These notes can be downloaded as an actual notebook, click the GitHub logo at the top of the page and choose .ipynb. The following is not runnabel in the notebook as is.

```
git pull
```

You can also go through these same basic steps: add, commit, push

2.3. More on git

- GitHub Hello World
- Software Carpentry Git Novice Lesson

Also, in Spring 2022, I'm teaching a section of CSC392: Topics in Computing, Introduction to Computer Systems, that will cover tools of the trade (git, bash, etc) and how they all work in great detail.

2.4. More on Python

Read Pep 8 to see what good style in Python is.

3. Getting help, object inspection, loading data

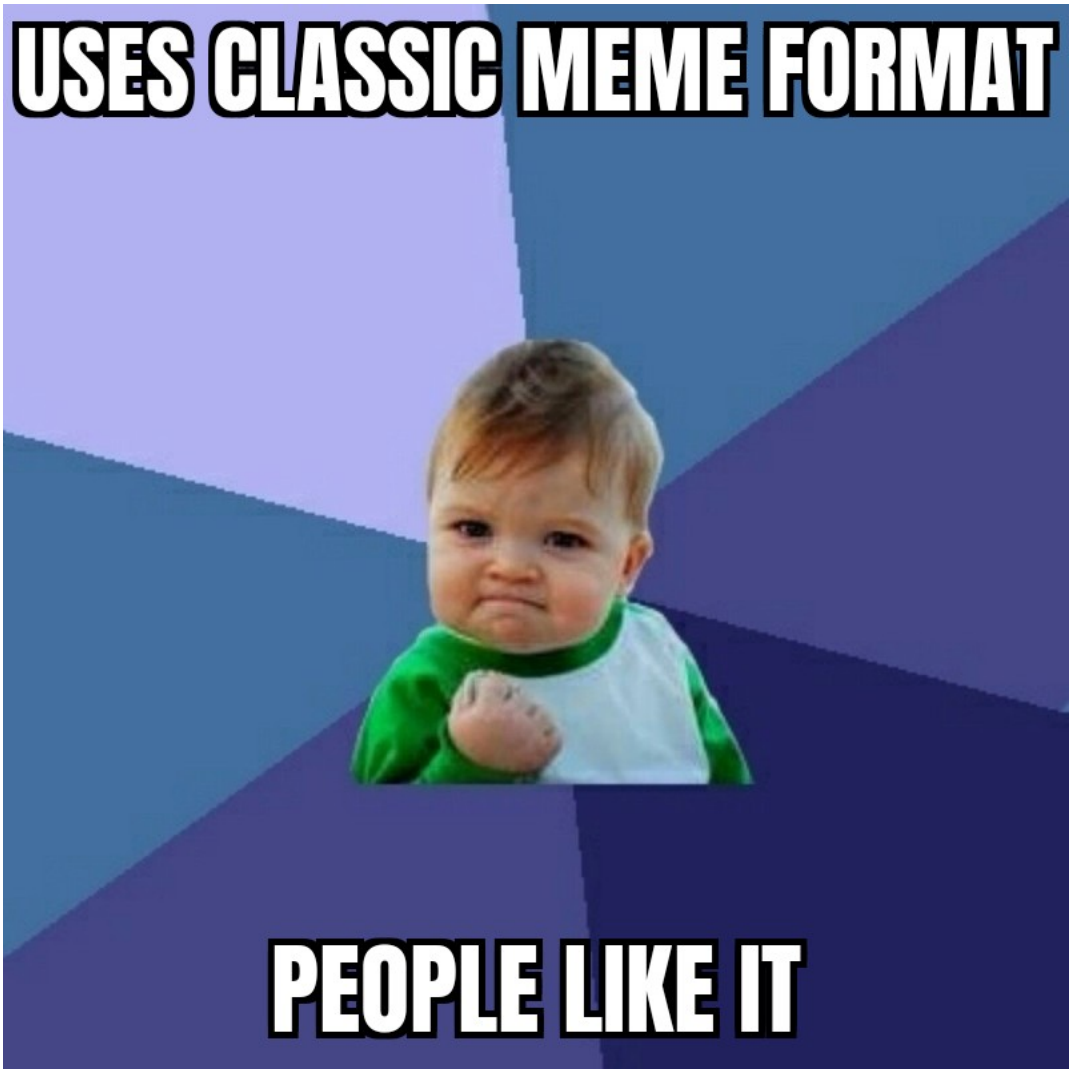
3.1. First, Don't Worry members

Class Response Summary:

funny CS memes

Hey, Don't even worry about it.

David Steele
@David_C_Steele
"We just left a rest stop! Why didn't you go then?"
SMITH



3.2. Getting Help in Jupyter

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

Given this code excerpt, how could you print out “Sarah_Brown”?

```
first = 'Sarah'
last = 'Brown'
```

We can use jupyter popup help wiht shift +tab or `?`

```
print?
```

Or the base python `help` function

```
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.
```

Notice that function can take multiple arguments and has a keyword argument (must be used like `argument=value`) described as `sep=' '`. This means that by default it adds a space

```
print(first,last)
```

```
Sarah Brown
```

But we can change the separator.

```
Sarah_Brown
```

Note that it also defaults to end to use `\n`

```
print(first,last)
print('hello')
```

```
Sarah Brown
hello
```

Where does this help information come from?

Note

You can copy code from the notes, try hovering over this

```python
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
```

We can apply `help` on the function we wrote

```python
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 (+/-)
```

It gets the docstring

3.3. Everything is an Object in Python

we can use the builtin function `type` to inspect them, and get attributes with `.`

```python
type(compute_grade)
```

```
function
```

```
compute_grade.__name__
```

```
'compute_grade'
```

```
c = 4.5
```

```
type(c)
```

```
float
```

```
c= 'hello'
```

```
type(c)
```

```
str
```

When do we use single vs double quotes?

- You can use either, unless you need to put one inside the string then use the other.

```
my_sentence = "The professor's name is Dr. Brown"
```

```
my_sentence = 'The professor's name is Dr. Brown'
```

```
  File "/tmp/ipykernel_1622/607286316.py", line 1
    my_sentence = 'The professor's name is Dr. Brown'
                                  ^
SyntaxError: invalid syntax
```

Yes we can escape special characters:

```
my_sentence = 'The professor\'s name is Dr. Brown'
```

but, it's less readable and not recommended.

3.4. Good Code is always relative

In programming for data science, we are often trying to tell a story.

> **Try it yourself**
>
> How might this goal change your code for this class relative to other code you have written or could imagine writing?

Python is a fully open source project and as such is governed by community standards and conventions.

> **Try it yourself**
>
> Find PEP8 (note that following it is part of earning python achievements)

The documentation for the full language is online too.

Guido van Rossum was the first main developer and wrote essays about python too.

it's pretty popular

3.5. Coffee Data

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

How was this dataset collected?

- reviewrs added to DB
- then scraped

Where did it come from?

- offee Quality Institute's trained reviewers.

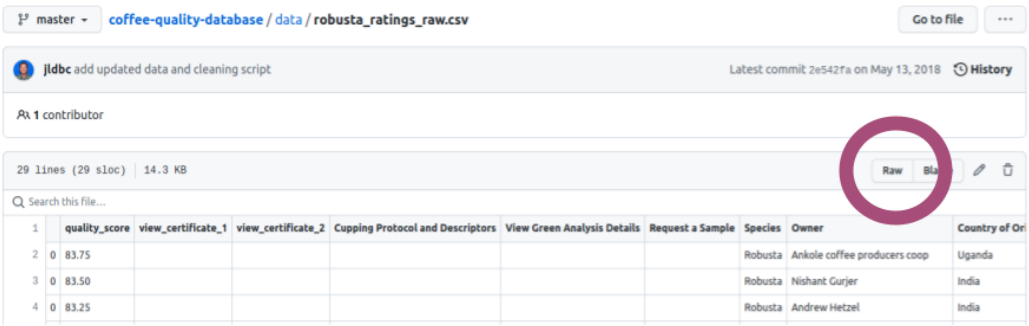
what format is it provided in?

- csv (Comma Separated Values)

what other information is in this repository?

- the code to scrape

Get raw url for the dataset click on the raw button on the [csv page](), then copy the url.



We'll save that url as a variable to work with it.

```
data_url = 'https://raw.githubusercontent.com/jldbc/coffee-quality-
database/master/data/robusta_data_cleaned.csv'
```

We will use a library called Pandas

```
import pandas as pd
# import library and give it an alias (nickname) pd
```

```
pd.read_csv(data_url)
```

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	M
0	1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ank cof produc
1	2	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	25	sethuram est
2	3	Robusta	andrew hetzel	India	sethuraman estate	NaN	N
3	4	Robusta	ugacof	Uganda	ugacof project area	NaN	uga
4	5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katu developm tr
5	6	Robusta	andrew hetzel	India	NaN	NaN	(s
6	7	Robusta	andrew hetzel	India	sethuraman estates	NaN	N
7	8	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	7	sethuram est
8	9	Robusta	nishant gurjer	India	sethuraman estate	RKR	sethuram est
9	10	Robusta	ugacof	Uganda	ishaka	NaN	nsubu un
10	11	Robusta	ugacof	Uganda	ugacof project area	NaN	uga
11	12	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	RC AB	sethuram est
12	13	Robusta	andrew hetzel	India	sethuraman estates	NaN	N
13	14	Robusta	kasozi coffee farmers association	Uganda	kasozi coffee farmers	NaN	N
14	15	Robusta	ankole coffee producers coop	Uganda	kyangundu coop society	NaN	ank cof produc coop uni
15	16	Robusta	andrew hetzel	India	sethuraman estate	NaN	N
16	17	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuram esta
17	18	Robusta	kawacom uganda ltd	Uganda	bushenyi	NaN	kawac
18	19	Robusta	nitubaasa ltd	Uganda	kigezi coffee farmers association	NaN	nitubaa
19	20	Robusta	mannya coffee project	Uganda	mannya coffee project	NaN	manr cof proj
20	21	Robusta	andrew hetzel	India	sethuraman estates	NaN	N
21	22	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuram esta
22	23	Robusta	andrew hetzel	United States	sethuraman estates	NaN	sethuram esta

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	M
23	24	Robusta	luis robles	Ecuador	robustasa	Lavado 1	our own
24	25	Robusta	luis robles	Ecuador	robustasa	Lavado 3	o laborat
25	26	Robusta	james moore	United States	fazenda cazengo	NaN	c cazen
26	27	Robusta	cafe politico	India	NaN	NaN	N
27	28	Robusta	cafe politico	Vietnam	NaN	NaN	N

28 rows × 44 columns

Try it yourself

Read the data in again, but with the index correct and save it to a variable.

Once we read it in, we van view the first 5 rows with the `head` method.

```
coffee_df.head()
```

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

5 rows × 43 columns

Important

Remember to comment & annotate your code

3.6. Follow Up questions

3.6.1. General Questions

How do you create code to scrape data from a website and compile it into a csv file?

Will we be using pandas a lot during the semester?

3.6.2. Clarifying

How do you auto finish your directories

How do you properly shut down Jupyter Notebook

Is pd some sort of variable we set or was it built in?

How should I be organized for this class? Keep it all in a single folder? Keep it on GitHub?

I'm still not sure how to keep everything together in a portfolio for the semester?

I am still wondering if I am using anaconda or just normal terminal

Can I push this code into my portfolio using the anaconda terminal

3.6.3. Grading Questions

How do we keep track of which achievements we've earned?

I don't really have many questions from today, but I was wondering if office hours were posted.

Will we always submit homework through the portfolio folder in github?

I'm just confused as how to view my feedback from the assignment

3.6.4. Questions we'll answer later this week

- does each column have a number assigned to it in data frames?
- Can other data types be imported into a notebook and edited the same way as .csv files?

3.7. More Practice

- How could you check if `pd` is built in or if we defined it?
- If we wanted to see more than 5 rows when printing the head of the dataset how would we do so?

> **Ram Token Opportunity**
>
> Contribute possible practice questions to the notes using the suggest an edit button behind the GitHub menu at the top of the page.

4. Pandas DataFrames

Today, we're going to explore DataFrames in greater detail. We'll continue using that same coffee dataset.

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

4.1. More about loading libraries

We can import pandas without the alias `pd` if we want, but then we have to use the full name everywhere

```python
import pandas
```

```python
pandas.read_csv(coffee_data_url)
```

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	M
0	1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ank cof produc
1	2	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	25	sethuram est
2	3	Robusta	andrew hetzel	India	sethuraman estate	NaN	N
3	4	Robusta	ugacof	Uganda	ugacof project area	NaN	uga
4	5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katu developm tr
5	6	Robusta	andrew hetzel	India	NaN	NaN	(s
6	7	Robusta	andrew hetzel	India	sethuraman estates	NaN	N
7	8	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	7	sethuram est
8	9	Robusta	nishant gurjer	India	sethuraman estate	RKR	sethuram est
9	10	Robusta	ugacof	Uganda	ishaka	NaN	nsubu un
10	11	Robusta	ugacof	Uganda	ugacof project area	NaN	uga
11	12	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	RC AB	sethuram est
12	13	Robusta	andrew hetzel	India	sethuraman estates	NaN	N
13	14	Robusta	kasozi coffee farmers association	Uganda	kasozi coffee farmers	NaN	N
14	15	Robusta	ankole coffee producers coop	Uganda	kyangundu coop society	NaN	ank cof produc coop uni
15	16	Robusta	andrew hetzel	India	sethuraman estate	NaN	N
16	17	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuram esta
17	18	Robusta	kawacom uganda ltd	Uganda	bushenyi	NaN	kawac
18	19	Robusta	nitubaasa ltd	Uganda	kigezi coffee farmers association	NaN	nitubaa
19	20	Robusta	mannya coffee project	Uganda	mannya coffee project	NaN	manr cof proj
20	21	Robusta	andrew hetzel	India	sethuraman estates	NaN	N
21	22	Robusta	andrew hetzel	India	sethuraman estates	NaN	sethuram esta
22	23	Robusta	andrew hetzel	United States	sethuraman estates	NaN	sethuram esta

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	M
23	24	Robusta	luis robles	Ecuador	robustasa	Lavado 1	our own
24	25	Robusta	luis robles	Ecuador	robustasa	Lavado 3	o laborat
25	26	Robusta	james moore	United States	fazenda cazengo	NaN	c cazer
26	27	Robusta	cafe politico	India	NaN	NaN	N
27	28	Robusta	cafe politico	Vietnam	NaN	NaN	N

28 rows × 44 columns

We'll use `pd` because that's the more common convention and so that we can type fewer characters throughout our code

```
import pandas as pd
```

4.2. Examining DataFrames

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

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

```
df.head()
```

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

5 rows × 43 columns

Using help, we can see that that head takes one parameter and has a default value of 5, which is why we got 5 rows, but we can get 2 instead

```
df.head(2)
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Num
1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	ankole coffee producers	
2	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	25	sethuraman estate	14/1148/2017

2 rows × 43 columns

We can look at the last rows with `tail`

```
df.tail(3)
```

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	C
26	Robusta	james moore	United States	fazenda cazengo	NaN	cafe cazengo	NaN	op
27	Robusta	cafe politico	India	NaN	NaN	NaN	14-1118-2014-0087	
28	Robusta	cafe politico	Vietnam	NaN	NaN	NaN	NaN	

3 rows × 43 columns

I told you this was a DataFrame, but we can check with type.

```
type(df)
```

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

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

```
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')
```

These are a special type called Index

```
type(df.columns)
```

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

It also has an index

```
df.index
```

```
Int64Index([ 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')
```

and values

```
df.values
```

```
array([['Robusta', 'ankole coffee producers coop', 'Uganda', ..., 1488.0,
        1488.0, 1488.0],
       ['Robusta', 'nishant gurjer', 'India', ..., 3170.0, 3170.0,
        3170.0],
       ['Robusta', 'andrew hetzel', 'India', ..., 1000.0, 1000.0, 1000.0],
       ...,
       ['Robusta', 'james moore', 'United States', ..., 795.0, 795.0,
        795.0],
       ['Robusta', 'cafe politico', 'India', ..., nan, nan, nan],
       ['Robusta', 'cafe politico', 'Vietnam', ..., nan, nan, nan]],
      dtype=object)
```

it also knows its own shape

```
df.shape
```

```
(28, 43)
```

we can use builtin fucntions on our DataFrame too not just its own methods and attributes.

```
len(df)
```

```
28
```

Why does `len` turn green? it's a python reserve word

4.3. Building a Data Frame programmatically

One way to build a data frame is from a dictionary:

```
people = {'names':['Sarah','Connor','Kenza'],
         'username':['brownsarahm','sudoPsych','kbdlh']}
```

```
people
```

```
{'names': ['Sarah', 'Connor', 'Kenza'],
 'username': ['brownsarahm', 'sudoPsych', 'kbdlh']}
```

```
type(people)
```

```
dict
```

```
people_df = pd.DataFrame(people)
people_df
```

	names	username
0	Sarah	brownsarahm
1	Connor	sudoPsych
2	Kenza	kbdlh

```
type(people['names'])
```

```
list
```

```
type(people)
```

```
dict
```

```
type({4,5,5})
```

```
set
```

```
{4,5,5}
```

```
{4, 5}
```

```
people['names']
```

```
['Sarah', 'Connor', 'Kenza']
```

```python
type(set(people['names']))
```

```
set
```

```python
unique_people = set(people['names'])
type(unique_people)
```

```
set
```

```python
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')
```

```python
for col in df.columns:
    print(col.split('.'))
```

```
['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']
```

```python
for key,value in people.items():
    print(key,':',value)
```

```
names : ['Sarah', 'Connor', 'Kenza']
username : ['brownsarahm', 'sudoPsych', 'kbdlh']
```

```
df['Owner']
```

```
1          ankole coffee producers coop
2                        nishant gurjer
3                         andrew hetzel
4                                ugacof
5          katuka development trust ltd
6                         andrew hetzel
7                         andrew hetzel
8                        nishant gurjer
9                        nishant gurjer
10                               ugacof
11                               ugacof
12                       nishant gurjer
13                        andrew hetzel
14    kasozi coffee farmers association
15         ankole coffee producers coop
16                        andrew hetzel
17                        andrew hetzel
18                  kawacom uganda ltd
19                       nitubaasa ltd
20                mannya coffee project
21                        andrew hetzel
22                        andrew hetzel
23                        andrew hetzel
24                          luis robles
25                          luis robles
26                          james moore
27                        cafe politico
28                        cafe politico
Name: Owner, dtype: object
```

```
df.Owner
```

```
1          ankole coffee producers coop
2                        nishant gurjer
3                         andrew hetzel
4                                ugacof
5          katuka development trust ltd
6                         andrew hetzel
7                         andrew hetzel
8                        nishant gurjer
9                        nishant gurjer
10                               ugacof
11                               ugacof
12                       nishant gurjer
13                        andrew hetzel
14    kasozi coffee farmers association
15         ankole coffee producers coop
16                        andrew hetzel
17                        andrew hetzel
18                  kawacom uganda ltd
19                       nitubaasa ltd
20                mannya coffee project
21                        andrew hetzel
22                        andrew hetzel
23                        andrew hetzel
24                          luis robles
25                          luis robles
26                          james moore
27                        cafe politico
28                        cafe politico
Name: Owner, dtype: object
```

```
df
```

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

	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.
24	Robusta	luis robles	Ecuador	robustasa	Lavado 1	our own lab	
25	Robusta	luis robles	Ecuador	robustasa	Lavado 3	own laboratory	
26	Robusta	james moore	United States	fazenda cazengo	NaN	cafe cazengo	
27	Robusta	cafe politico	India	NaN	NaN	NaN	14-11:
28	Robusta	cafe politico	Vietnam	NaN	NaN	NaN	

28 rows × 43 columns

Key points:

write three things to remember from today's class

4.4. Questions After Classroom

many overlapping questions today

4.5. General

How to know which function to use in certain problems or situations

4.6. Clarifying

Is there a way to have a set show the duplicates that get discarded?

being able to access the code somewhere without asking to scroll would be nice

4.7. Course Admin

When will homeworks be posted/due typically?

4.8. Questions we'll answer later

can you use cast a pandas dataframe into a set?

4.9. Try it yourself

- Create variables of three different types with facts about yourself. Use descriptive variable names relative to the contents, not their types.

- Create a list, again with a descriptive name, and print out the types

```
<class 'str'>
<class 'int'>
<class 'list'>
```

- Write a function, `type_extractor` that takes a list and a type and returns the item of that type from the list
- Test your function on all three items from your dictionary.
- Use one type of jupyter help on your function, what does it display? If it doesn't display anything modify your function so that help will work.
- Make yourself notes in the most memorable way for you about what a DataFrame is.

Ram Token Opportunity

Contribute possible practice questions to the notes using the suggest an edit button behind the GitHub menu at the top of the page.

5. More Loading Data, Indexing, and Iterables

As always, we'll start with loading pandas.

```
import pandas as pd
```

5.1. Checking in on help hours

if you missed class, check over the office hours schedule and e-mail if you can or cannot attend at least one time

5.2. Portfolio Preparation and Maintainance

We'll spend a little time today getting your portfolio ready for the first check.

5.2.1. Access your portfolio

Go to your portflio

- from the course organization
- from the list of your recent repositories on the left hand side of the GitHub home page

optionally, open it locally as well (we're going to update content and)

5.2.2. Start your Know, Want to Know, Learned Table

In each portfolio submission introduction, you'll reflect on what you've learned. To get ready for that, we'll first make note of what you already know and what you want to know.

1. edit `submission_1_intro` in your portfolio locally or on GitHub:
2. In the KWL section in the first two bullets after each skill with what you know and want to know. You can edit these in more detail later.

This will render as a table in your built portflio, for reference on the syntax, refer to the Tables section of the jupyterbook Myst Markdown cheatsheet.

Warning

If you work on this in the GitHub website, be sure to pull these chances locally before you start working offline next

5.2.3. Merge the setup work

Once you're done, Go to your pull request tab, and select the feedback Pull Reques. Commit any suggestions if you'd like and then merge the PR.

Warning

only do this after grading

Note

To view the feedback, after merging the PR, remove `is:open` from the search bar on the PR page

5.3. Indexing

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

What will topics[-1] return?

```
topics[-1]
```

```
'pandas'
```

Using negative indices starts from the right. The last element is -1. The first is 0.

5.4. Reading DataFrames from Websites

We'll first read from the course website.

```
course_comms_url =
'https://rhodyprog4ds.github.io/BrownFall21/syllabus/communication.html'
```

So far, we've read data in from a .csv file with `pd.read_csv` and created a DataFrame with the constructor `pd.DataFrame` using a dictionary. Pandas provides many interfaces for reading in data. They're described on the Pandas IO page.

We can use the `read_html` method to read from this page. We know that it has multple tables on the page, so lets see what it t does:

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

Note

Using the documentation for a library (and the base language) is totally expected and normal part of programming. That's what you should use as your primary source for questions in this class. Other sources can become outdated pretty quickly as the language changes, but most of the libraries we'll use have processes in place to ensure that their own documentation gets updated at the same time the code does.

Warning

If you use other sources and get advised to solutions that are deprecated you may not earn achievements for that work.

```
[              Day                Time             Location      Host
 0          Monday    1:00:00 PM-2:30     inperson roomtbd  Chamundi
 1       Wednesday         4:00:00 PM     inperson roomtbd  Chamundi
 2        Wedneday       2:00:00 PM-3     inperson roomtbd  Chamundi
 3       Wednesday    7:00:00 PM-8:30          gather.town     Sarah
 4          Friday  5:00:00 PM-6:30pm          gather.town  Chamundi
 5  By appointment                 TBD  in person Tyler 134     Sarah,
                                                usage platform  \
 0                                       in class  prismia
 1                                       any time  prismia
 2                                       any time  prismia
 3          private questions to your assignment   github
 4    for general questions that can help others   github
 5                            to share resources   github
 6  matters that don't fit into another category   e-mail

                       area                                            note
 0                     chat  outside of class time this is not monitored cl...
 1            message board                          for discussion with peers
 2       download transcript  use after class to get preliminary notes eg if...
 3  issue on assignment repo                              eg bugs in your code"
 4   issue on course website  eg what the instructions of an assignment mean...
 5   pull request on website       remember to request ram tokens if applicable
 6    to brownsarahm@uri.edu  remember to include `[CSC310]` or `[DSP310]` (...  ,
                                          usage                     area  \
 0  matters that don't fit into another category  to brownsarahm@uri.edu

                                                 note
 0  remember to include `[CSC310]` or `[DSP310]` (...  ,
                                          usage                     area  \
 0        private questions to your assignment  issue on assignment repo
 1  for general questions that can help others   issue on course website
 2                            to share resources   pull request on website

                                                 note
 0                               eg bugs in your code"
 1  eg what the instructions of an assignment mean...
 2       remember to request ram tokens if applicable  ,
       usage                area  \
 0  in class                chat
 1  any time        message board
 2  any time  download transcript

                                                 note
 0  outside of class time this is not monitored cl...
 1                            for discussion with peers
 2  use after class to get preliminary notes eg if...  ]
```

It appears to have read all of them, lets check the type:

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

```
list
```

Since we know it's a list, we'll save it to a variable that indicates that.

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

If we get just the first element,

```
type(comms_list[0])
```

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

it's a DataFrame and prints accordingly.

```
comms_list[0]
```

	Day	Time	Location	Host
0	Monday	1:00:00 PM-2:30	inperson roomtbd	Chamundi
1	Wednesday	4:00:00 PM	inperson roomtbd	Chamundi
2	Wedneday	2:00:00 PM-3	inperson roomtbd	Chamundi
3	Wednesday	7:00:00 PM-8:30	gather.town	Sarah
4	Friday	5:00:00 PM-6:30pm	gather.town	Chamundi
5	By appointment	TBD	in person Tyler 134	Sarah

Since it's a list, we can use base python's `len` function to check how many tables there are

```
len(comms_list)
```

```
5
```

We've seen the first table and know it's the help hours, so we can save that to a separate variable and use it

```
help_df = comms_list[0]
```

We've inspected the dataframe some before, but we can also check the type of each column.

```
help_df.dtypes
```

```
Day         object
Time        object
Location    object
Host        object
dtype: object
```

Further Reading

You can read more about the details of data types in Pandas in the documentation

5.5. How are objects printed in jupyter?

Question from class

Q: Why does it have `dtype:object` after the type for each row? A: the last line is information about the object that is being printed out.

To understand this, let's save the thing we're curious to a variable so we can examine it multiple ways more easily.

```
help_df_types = help_df.dtypes
```

Note

this section is added, it didn't happen this way in class. This section, describes **how** I figured out the answer to the question about why that extra line is diplayed.

Next we'll check the type of this object and its shape

```
type(help_df_types)
```

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

a Series is like a DataFrame, but just one row with headings, and then rotated.

```
help_df_types.shape
```

```
(4,)
```

This means that it's length is 4 and it's a 1 dimensional object; the column headers have converted to an index and are treated as metadata, but not a part of the actual data.

So, the line we're interested in is not a part of the object, because it's length 4 and the thing we're curious about is the fifth line.

We'll pick one variable from the DataFrame and check its type

```
type(help_df['Day'])
```

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

This is also a Series, so lets check its output

```
help_df['Day']
```

```
0            Monday
1         Wednesday
2          Wedneday
3         Wednesday
4            Friday
5    By appointment
Name: Day, dtype: object
```

THe last line of this one is information about the Series, its name, and its dtype.

Let's make another series, and see how it prints

```
pd.Series([5,4,5])
```

```
0    5
1    4
2    5
dtype: int64
```

The last line is the dtype of the Series; so in our original object, that last line is because the list of dtypes is the of type object.

```
help_df_types
```

```
Day         object
Time        object
Location    object
Host        object
dtype: object
```

5.6. How do we know what to check?

we examined the DataFrame so far by (me) knowing what to look for.

In python objects you can progrmamatically find what to look for with the `__dict__` attribute or we can rely on the [online documentation]() or use it via help.

In ipython (what we use in jupyter, by default) we can use the `?` for help

```
pd.DataFrame?
```

```
help(pd.DataFrame)
```

Everything is Data

writing good documentation lets people who use your code get help for free Not only do help tools use the docs, but that website is generated programmatically using a tool called Sphinx from the documentation inside the code. You can access the docstring of a python using the `.__doc__` attribute

Caution

[ipython help]() read about how it works, if it doesn't work for you to try to figure out why

```
Help on class DataFrame in module pandas.core.frame:

class DataFrame(pandas.core.generic.NDFrame, pandas.core.arraylike.OpsMixin)
 |  DataFrame(data=None, index: 'Axes | None' = None, columns: 'Axes | None' = None,
dtype: 'Dtype | None' = None, copy: 'bool | None' = None)
 |
 |  Two-dimensional, size-mutable, potentially heterogeneous tabular data.
 |
 |  Data structure also contains labeled axes (rows and columns).
 |  Arithmetic operations align on both row and column labels. Can be
 |  thought of as a dict-like container for Series objects. The primary
 |  pandas data structure.
 |
 |  Parameters
 |  ----------
 |  data : ndarray (structured or homogeneous), Iterable, dict, or DataFrame
 |      Dict can contain Series, arrays, constants, dataclass or list-like objects. If
```

```
 |      data is a dict, column order follows insertion-order.
 |
```

Try it yourself!

Make a list of the shape of all of the tables on the syllabus Achievements page.

```
achievements_url =
'https://rhodyprog4ds.github.io/BrownFall21/syllabus/achievements.html'
```

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

This solution uses a list comprehension which allows us to compress a loop. It's equivalent to the following with a for loop

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

5.7. Lambdas and Dictionaries for switching

What if we want to print out the first column for the dataFrame if it has more than 3 columns and the whole thing if it has 3 or less columns?

Two ways of writing a function

```
# with the def key
def first_col_f(d):
    return d[d.columns[0]]

# lambda (anonymous function)
first_col_l = lambda d: d[d.columns[0]]

first_col_f(help_df) == first_col_l(help_df)
```

```
0    True
1    True
2    True
3    True
4    True
5    True
Name: Day, dtype: bool
```

Try it yourself

read the code excerpt above carefully and try to match up the parts of a function: its name, the parameter list, and the body. Try writing your own lambda function.

Lambdas are an example of an anonymous function

We can put functions in dictionaries, or even define a lambda right in the dictionary.

```
df_display = {True: lambda d: d[d.columns[0]],
       False: lambda d: d}

for df in pd.read_html(achievements_url):
    _, n_cols = df.shape
    print(df_display[n_cols>3](df))
```

Question from class

Python does have ternary operators but the dictionary is a more common way to achieve this and goal and more common patterns are better for readability

Further Reading

You can refer to the official documentation on lambda functions for a brief syntax description or this tutorial on Real Python for more context, history, and examples.

Important

We'll see lambdas again and again. Starting with summarizing data and again when cleaning. Understanding how to use these will help.

```
   Unnamed: 0_level_0                                            topics  \
                 week                                Unnamed: 1_level_1
0                   1                            [admin, python review]
1                   2                       Loading data, Python review
2                   3                         Exploratory Data Analysis
3                   4                                     Data Cleaning
4                   5                      Databases, Merging DataFrames
5                   6  Modeling, Naive Bayes, classification performa...
6                   7                   decision trees, cross validation
7                   8                                        Regression
8                   9                                        Clustering
9                  10                             SVM, parameter tuning
10                 11                             KNN, Model comparison
11                 12                                     Text Analysis
12                 13                                   Images Analysis
13                 14                                     Deep Learning

                            skills
                 Unnamed: 2_level_1
0                           process
1      [access, prepare, summarize]
2            [summarize, visualize]
3   [prepare, summarize, visualize]
4    [access, construct, summarize]
5        [classification, evaluate]
6        [classification, evaluate]
7            [regression, evaluate]
8            [clustering, evaluate]
9                 [optimize, tools]
10                 [compare, tools]
11                   [unstructured]
12            [unstructured, tools]
13                 [tools, compare]
0             python
1            process
2             access
3          construct
4          summarize
5          visualize
6            prepare
7     classification
8         regression
9         clustering
10          evaluate
11          optimize
12           compare
13      unstructured
14          workflow
Name: (Unnamed: 0_level_0, keyword), dtype: object
0             python
1            process
2             access
3          construct
4          summarize
5          visualize
6            prepare
7     classification
8         regression
9         clustering
10          evaluate
11          optimize
12           compare
13      unstructured
14          workflow
Name: (Unnamed: 0_level_0, keyword), dtype: object
0             python
1            process
2             access
3          construct
4          summarize
5          visualize
6            prepare
7     classification
8         regression
9         clustering
10          evaluate
11          optimize
12           compare
13      unstructured
14          workflow
Name: (Unnamed: 0_level_0, keyword), dtype: object
```

In that excerpt df_display has a key that is defined to be true or false. The value for each item in the dictionary (separated by commas `,`) is a lambda fucntion, both of which take a parameter `d`, and one of which returns the first column `d[d[columns[0]]` and the other fow which returns the whole data frame `d`.

In thee loop, we set index into the dictionary with the key `n_cols > 3` with `df_display[n_cols>3]` sometimes it will be true and other times it will be false, which matches the two keys defined in the dictionary. Then the parameter it takes is `d`, which we pass the whole data frame `df` with the `(df)` at the end of the line.

> **Try it Yourself!**
>
> What does the _ do?

```
Try using that dictionary outside of the loop.  What is the value for a give key
if you print it out like `df_display[ a_key_value]`? What if you use `True` or
`False` directly? what if you try a number? What is the type of that? What if you
pass something that's not a DataFrame as the parameter?  Make notes about these outputs
```

5.8. Questions after class

> **Note**
>
> add a question with a pull request; earn 1-2 ram tokens for submitting a question with the answer (with sources)

5.9. More Practice

- What `type` is the shape of a `pandas.DataFrame`?
- use a list comprehension to create a list that you could use as column names for data that consists of `N` measurements. Set `N=5` for now, but you suspect that the number might change.

- create a list of items with different types, then Create a dictionary with the types as keys using a dictionary comprehension. Dictionary comprehensions are similar to list comprehensions, in their form.

- Create a `lambda` function to print return the first 2 rows of a data frame

> **Ram Token Opportunity**
>
> Contribute possible practice questions to the notes using the suggest an edit button behind the GitHub menu at the top of the page.

6. Exploratory Data Analysis

```
import pandas as pd
```

6.1. Data Frame from lists of Lists

On Friday, we collectively made a list of strings

```
# %load http://drsmb.co/310read

# share a sentence or a few words about how class is going or
# one takeaway you have from class so far.
#  and attribute with a name affter a hyphen(-)
# You can remain anonymous (this page & the notes will be fully public)
# by attributing it to a celebrity or psuedonym, but include *some* sort of attribution
sentence_list = [
'Programming is a Practice. - Dr. Sarah Brown',
"So far it's going pretty good. - Matt",
'Pretty good pace, Github is still a little confusing is all - A',
"Class is going well, I'm really excited for the new skills that I will attain through
this course. - Diondra",
'Good straight forward - Adam',
'Good pace and engaging - Jacob',
'I like cheese - Anon',
'Going well, I enjoy python - Aiden',
"Class is going very well, I'm excited to learn more about manipulating data sets -
Greg",
'Really enjoying the class so far, Clear instructions and outcomes - Michael',
'So far this class is going well, very engaging lectures. - Anon',
'Great pace and notes have been really useful - Brandon',
'Good pace and easy to follow - Muhammad',
'Class is going well and engaging, but also a little difficult getting into the swing of
things - Isaiah',
'Well paced, informative, and helpful - Vinnie',
'Spectacular -Michael Jackson',
'Very interesting! I am enjoying it a lot so far. Getting to experience pandas as well
as using github/jupyter has been cool. One thing I would change though is slowing down
the pace a bit. - Max'
]
```

We can check that by using type, first on the whole thing

```
type(sentence_list)
```

```
list
```

And then we can index into the list. Lists can be indexed by integers:

```
sentence_list[4]
```

```
'Good straight forward - Adam'
```

is one item from the list and then check its type:

```
type(sentence_list[4])
```

```
str
```

First, we'll convert our list of strings, to a list of lists with a list comprehension. This will iterate over each string in that list and apply the string method `split`. Since we passed the parameter `'-'`, it will split at that character.

```
[sent_attr.split('-') for sent_attr in sentence_list]
```

Question From Class

The split method will split at every occurence of the character passed.

Try it Yourself

Try splitting based on a different character (eg `' '`). What happens?

```
[['Programming is a Practice. ', ' Dr. Sarah Brown'],
 ["So far it's going pretty good. ", ' Matt'],
 ['Pretty good pace, Github is still a little confusing is all ', ' A'],
 ["Class is going well, I'm really excited for the new skills that I will attain
through this course. ",
  ' Diondra'],
 ['Good straight forward ', ' Adam'],
 ['Good pace and engaging ', ' Jacob'],
 ['I like cheese ', ' Anon'],
 ['Going well, I enjoy python ', ' Aiden'],
 ["Class is going very well, I'm excited to learn more about manipulating data sets ",
  ' Greg'],
 ['Really enjoying the class so far, Clear instructions and outcomes ',
  ' Michael'],
 ['So far this class is going well, very engaging lectures. ', ' Anon'],
 ['Great pace and notes have been really useful ', ' Brandon'],
 ['Good pace and easy to follow ', ' Muhammad'],
 ['Class is going well and engaging, but also a little difficult getting into the swing
of things ',
  ' Isaiah'],
 ['Well paced, informative, and helpful ', ' Vinnie'],
 ['Spectacular ', 'Michael Jackson'],
 ['Very interesting! I am enjoying it a lot so far. Getting to experience pandas as
well as using github/jupyter has been cool. One thing I would change though is slowing
down the pace a bit. ',
  ' Max']]
```

If we save it to a variable, we can analyze it better, for example indexing it

```
list_of_lists = [sent_attr.split('-') for sent_attr in sentence_list]
list_of_lists[0]
```

```
['Programming is a Practice. ', ' Dr. Sarah Brown']
```

This is a list, which we can check with:

```
type(list_of_lists[0])
```

```
list
```

If we take one item from that, it's a string

```
list_of_lists[0][1]
```

```
' Dr. Sarah Brown'
```

The list of lists is the same length as our original sentence_list, because it was made from that.

```
len(sentence_list)
```

```
17
```

```
len(list_of_lists)
```

```
17
```

Then we can pass our list of lists to the DataFrame constructor and set the column headings with the `columns` parameter, which accepts a list.

```
pd.DataFrame([sent_attr.split('-') for sent_attr in sentence_list],
             columns=['sentence','attribution'])
```

Question From Class

While the list comprehension iterate over each string in that list, because it's a list, in order to index into it, we have to use an integer.

Iterating is taking each member in an object in turn, indexing is picking out a specified item. Iterating typically is done with the `for` keyword (either in a loop or a comprehension), indexing is done with `[]`

	sentence	attribution
0	Programming is a Practice.	Dr. Sarah Brown
1	So far it's going pretty good.	Matt
2	Pretty good pace, Github is still a little con...	A
3	Class is going well, I'm really excited for th...	Diondra
4	Good straight forward	Adam
5	Good pace and engaging	Jacob
6	I like cheese	Anon
7	Going well, I enjoy python	Aiden
8	Class is going very well, I'm excited to learn...	Greg
9	Really enjoying the class so far, Clear instru...	Michael
10	So far this class is going well, very engaging...	Anon
11	Great pace and notes have been really useful	Brandon
12	Good pace and easy to follow	Muhammad
13	Class is going well and engaging, but also a l...	Isaiah
14	Well paced, informative, and helpful	Vinnie
15	Spectacular	Michael Jackson
16	Very interesting! I am enjoying it a lot so fa...	Max

Hint

We built the list of lists here with a list comprehension because it's only a few items, but for a list of longer lists, you might use a for loop and `append`

6.2. Summarizing Data

```
coffee_data_url = 'https://raw.githubusercontent.com/jldbc/coffee-quality-database/master/data/robusta_data_cleaned.csv'
coffee_df = pd.read_csv(coffee_data_url)
```

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.

```
coffee_df.head()
```

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mi
0	1	Robusta	ankole coffee producers coop	Uganda	kyangundu cooperative society	NaN	anko coffe produce
1	2	Robusta	nishant gurjer	India	sethuraman estate kaapi royale	25	sethurama estat
2	3	Robusta	andrew hetzel	India	sethuraman estate	NaN	Na
3	4	Robusta	ugacof	Uganda	ugacof project area	NaN	ugac
4	5	Robusta	katuka development trust ltd	Uganda	katikamu capca farmers association	NaN	katuk developme tru

5 rows × 44 columns

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

```
coffee_df.describe()
```

	Unnamed: 0	Number.of.Bags	Harvest.Year	Fragrance...Aroma	Flavor	Aftertaste
count	28.000000	28.000000	28.000000	28.000000	28.000000	28.000000
mean	14.500000	168.000000	2013.964286	7.702500	7.630714	7.559643
std	8.225975	143.226317	1.346660	0.296156	0.303656	0.342469
min	1.000000	1.000000	2012.000000	6.750000	6.670000	6.500000
25%	7.750000	1.000000	2013.000000	7.580000	7.560000	7.397500
50%	14.500000	170.000000	2014.000000	7.670000	7.710000	7.670000
75%	21.250000	320.000000	2015.000000	7.920000	7.830000	7.770000
max	28.000000	320.000000	2017.000000	8.330000	8.080000	7.920000

8 rows × 21 columns

We can also select one variable at a time

```
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
```

To dig in on what the quantiles really mean, we can compute one manually.

First, we sort the data, then for the 25%, we select the point in index 6 because becaues there are 28 values.

```
balance_sorted = coffee_df['Balance'].sort_values().values
balance_sorted[6]
```

```
7.5
```

We can also extract each of those statistics individually, by name. The quantiles are tricky, we can't ust `.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(.25)
```

```
7.5600000000000005
```

We can also pass other values

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

```
7.83
```

Calculate the mean of the `Aftertaste` column:

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

```
7.559642857142856
```

6.3. What about the nonnumerical variables?

For example the color

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

```
0    Green
1      NaN
2    Green
3    Green
4    Green
Name: Color, dtype: object
```

We can get the prevalence of each one with `value_counts`

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

```
Green           20
Blue-Green       3
Bluish-Green     2
None             1
Name: Color, 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?

What country is most prevalent in this dataset?

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

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

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

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

```
'India'
```

Question From Class

Q: Can we calculate the mode to find the most prevalent? A: Yes. We can also use the mode function, which works on both numerical or nonnumerical values.

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

```
0    India
dtype: object
```

6.4. Questions After Class

6.4.1. General Questions

How to know what functions are compatible with other functions

Best place to find all the individual functions based on the .describe() function

6.4.2. Clarifying

Is sent_attr in the DataFrame loop its own variable that we assign, or is it a base Python function?

Difference between %load and how we've been importing links to datasets in previous classes?

When I ran [sent_attr.split('-') for sent_attr in sentence_list] in Jupyter, I got a nameerror

Why do we use the value quantile instead of using quartile since we can use mean to find mean?

6.4.3. Coure Admin and Assignment Questions

when are the achievements we earn in class going to be inputted in brightspace?

is there anything we have to do for the assignment after commiting the changes to

When we push the homework are we pushing it to the portfolio?

are there panda functions to read files other than CSV?

Still a little unsure about what the num_numerical specifically is, is it just the number of numerical columns?

6.5. More Practice

1. Produce a table with the mean for each score.
2. Which variables have the most missing data?
3. What's the total number of bags of coffee produced?
4. Which ratings have similar ranges? (max, min)
5. Which rating are most consistent across coffees?
6. What score cutoff could you apply in order to select the top 10 best for Balance?

1. Portfolio Setup, Data Science, and Python

Due: 2020-09-12

1.1. Objective & Evaluation

This assignment is an opportunity to earn level 2 achievements for the `process` and `python` and confirm that you have all of your tools setup, including your portfolio.

1.2. To Do

> **Important**
>
> If you have trouble, check the GitHub FAQ on the left before e-mailing

```
```{warning}
If you have trouble with the (*)d steps, don't worry, we can help work around these later. To help
us out, document the errors as bugs on your repository.
```
```

Your task is to:

1. Install required software from the Tools & Resource page
2. Create your portfolio, by accepting the assignment
3. Learn about your portfolio from the README file on your repository.
4. edit `_config.yml` to set your name as author and change the logo if you wish
5. Fill in `about/index.md` with information about yourself(not evaluated, but useful) and your own definition of data science (graded for **level 1 process**)
6. (*) Install some additional python packages with: `pip install pip install -r requirements.txt` (this is a python operation, so use anaconda prompt on Windows, if the pip version doesn't work, try it with conda: `conda install --file requirements.txt`) form inside the portfolio folder
7. (*) Configure precommit to help keep your repo clean with `pre-commit install`. If this step doesn't work, see the portfolio README under "Using your Jupyter Book Portfolio"
8. Add a Jupyter notebook called `grading.ipynb` to the `about` folder and write a function that computes a grade for this course, with the following docstring. Include:
   - 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.
   - 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
9. Add the line `- file: about/grading` in your `_toc.yml` file.

> **Note**
>
> If you get stuck on any of this after accepting the assignment and creating a repository, you can create an issue on your repository, describing what you're stuck on and tag us: @rhodypro4dg/fall21instructors
>
> To do this click Issues at the top, the green "New Issue" button and then type away.

> **Important**
>
> remember to add, commit, and push your changes so we can see them

```
'''
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 (+/-)
'''
```

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-'

assert compute_grade(4,3,1) == 'D'

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

> **Warning**
>
> your function can have a different name than `compute_grade`, but make sure it's your function name, with those parameter values in your tests.

> **Note**
>
> when the value of the expression after `assert` is `True`, it will look like nothing happened. `assert` is used for testing

1.3. Submission Instructions

Create a Jupyter Notebook with your function in your portfolio folder commit and push the changes.

In your browser, view the `gh-pages` branch to see your compiled submission, as `portfolio.pdf` or by viewing your website.

There will be a pull request on your repository that is made by GitHub classroom, request a review from @rhodypro4dg/fall21instructors.

2. Practicing Python and Accessing Data

due : 2020-09-21

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`.

In this assignment, you'll practice/ review python skills by manipulating datasets and extracting. The following table summarizes the grading. It supplements the skill definitions from the Achievement Definitions.

Task	Skills (max level)
identify possible uses for data in a data science pipeline	[process (1)]
load data from one file format	[access (1)]
load data from at least two of (.csv, .tsv, .dat, database, .json)	[access (2)]
compare the data formats	[access (2)]
complete the assignment in python	python (1)]
use python data types (eg dictionaries) to prepare information about datasets	[python (2)]
use informative variable names, pythonic iteration, and other common PEP 8 conventions	[python (2)]
display DataFrame properties	[summarize (1)]

Table 2.1 rubric for grading

First, accept the assignment . It contains a notebook with some template structure (and will set you up for grading).

2.2. 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 notebook that includes:

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

Hint

The Datasets page has information about data for any assignment. For this assignment, the Best for loading directly into a notebook section is probably the best place to start.

2.3. Store them for loading

Create a list of dictionaries in `datasets.py`, so that there is one dictionary for each dataset with the url, a name, and what function should be used to load the data into a `pandas.DataFrame`.

Hint

Any `.py` file can become a module

Hint

The goal here is to set up this list of dictionaries so that you can load data using different functions in each pass through the loop, without an `if` statement. You'll be iterating over the list of dictionaries, so the loop variable be a different dictionary each time.

see the where we used a lambda in a dictionary in [the class notes]() for more information.

Tip

Urls are strings. The `string` class in python has a lot of helpful methods for manipulating strings, like `split`.

Tip

The [pandas IO]() page has information about how to read data in and save data out of pandas.

2.4. Make a dataset about your datasets

Import the list from the `datasets` module you created in the step above. Then [iterate]() over the list of dictionaries, and:

1. 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.
2. record attributes about the dataset as in the table below in a list of lists:
3. Use that to create a DataFrame with the following columns:

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

Table 2.2 Meta Data Description of the DataFrame to build

2.5. Manipulate your datasets

For one dataset that includes nonnumerical data:

- display the heading and the last 4 rows
- make and display a new data frame with only the numerical columns (select these programmatically)

For any other dataset:

- display the heading and the first three 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:

- display the first 3 odd rows (eg 1,3,5) of the data for two columns of your choice

2.6. Exploring data files

For each dataset, in a separate section of your notebook titled `When things go wrong`:

- try reading in data with the wrong `read_` function and make notes about what happens.
- was the format that the data was provided in a good format? why or why not?
- 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 manualy (any text editor, including jupyter can edit a `.csv`), making notes about what changes you made in a markdown cell.

Think Ahead

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 eg this google scholar result How could something like this impact your work as a data scientist?

Warning

This section is not required, but is intended to help you get started thinking about ideas for your portflio. 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.

Portfolio Dates and Key Facts

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipykernel_1715/311973014.py in <module>
      5
      6 rubric_df = pd.concat([rubric_df,
----> 7                        assignment_dummies,
      8                        portfolio_dummies],axis=1)
      9

NameError: name 'assignment_dummies' is not defined
```

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

Current

Check 1

The first portfolio check will be due October 15

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipykernel_1715/2819809228.py in <module>
----> 1 portfolio_df[portfolio_df['P1']==1]

NameError: name 'portfolio_df' is not defined
```

Upcoming Checks

Check 2: November 12 Check 3: December 5 Check 4: December 20

Submission Introductions

Your portfolio will be assessed both on your demonstration of skills through your chapters that are inspired by the prompts and that your summary demonstrates that you *know* you learned the skills.

Each portfolio submission, you will edit the corresponding `submission_x_intro.md` file. This is where you write your summary and reflect on your learning. This reflection does not need to be long, it shouldn't take a very long time. It's okay for it to be brief, mostly bullet points.

One part of your introduction is a **KWL** or **K**now, **W**ant to know, **L**earned, chart.

In your file it will look like this:(but longer)

```
```{list-table} Portfolio 1 KWL Chart
:header-rows: 1
:name: kwl1

* - Skill
  - Know
  - Want to Know
  - Learned
* - python
  - basics
  - more efficient types
  -
* - access
  - that datasets are collated on kaggle
  - how to load data for analysis
  - how to load data from 3 different types and compare them
* - ...
  - ...
  - ...
  - ...
```
```

and when you build your portfolio it will render like this:

Skill	Know	Want to Know	Learned
python	basics	more efficient patterns	pep8 patterns and common conventions,
access	that datasets are collated on kaggle	how to load data for analysis	how to load data from 3 different types and compare them
…	…	…	…

Table 1 Portfolio 1 KWL Chart

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

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

```
repos:
-   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 run --all-files
```

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 bold**` in either case

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)
```

Configurations

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.

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 Beibhinn (beibhinn)
- via Prismia.chat during class
- by creating an issue

Syllabus FAQ

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

Can I submit this assignment late if …?

Git and GitHub

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

My command line says I cannot use a password

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

My portfolio won't compile

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

Code Errors

Key Error

<bound method

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.

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.

DataFrame
: a data structure provided by pandas for tabular data in python.

dictionary
: a mapping array that matches keys to values.

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 iterate over some data structure and typically uses the `for` keyword

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

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

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

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

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
```

```
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.

Requires 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

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

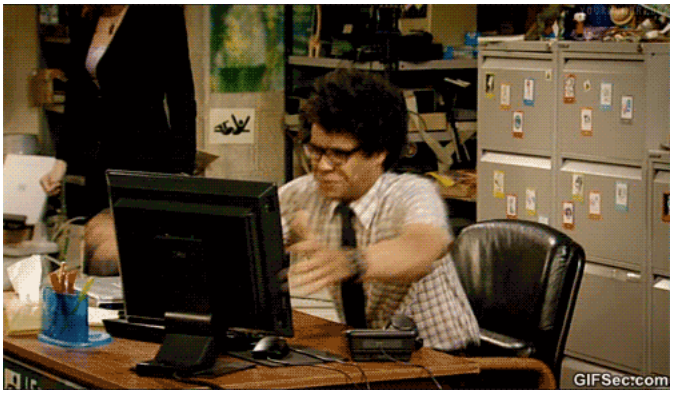
This section is for materials that are not specific to this course, but are likely useful. They are not generally required readings or installs, but are options or advice I provide frequently.

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 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.

A new book that might be of interest if you find programming classes hard is the Programmers Brain As of 2020-09-07, it is available for free by clicking on chapters at that linked table of contents section.

> **Where are your help tools?**
>
> In Python and Jupyter notebooks, what help tools do you have?

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.

There will be additional drills posted to Prismia chat for you to try.

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.

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.

> **Note**
>
> A fun version of this is rubber duck debugging

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.

To understand what the traceback is, how to read one, and common examples, see this post on Real Python.

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.

> **Note**
>
> We know that we don't currently teach a lot of this in our department, so in Spring 22 I'm teaching a brand new course on Computer Systems, that will help you understand the underlying concepts that make all of this stuff make sense, instead of just following recipes and debugging here and there.

If, for example, you come to me in week 5 and have never got an any environment working and you're trying for the first time, your grade will be hurt because you will be very far behind at that point. Ask for help early and often.

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

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.

By Professor Sarah M Brown
© Copyright 2021.