# Syllabus

## Contents

Syllabus

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

Welcome to CSC/DSP310: Programming For Data Science.

In this syllabus you will find an overview of the course, information about your instructor, course policies, restatements of URI policies, reminders of relevant resources, and a schedule for the course.

## About

### About the topic

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

### About the goals and preparation

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

Basic programming skills (CSC201 or CSC211) are a prerequisite to this course. This course is a prerequisite course to machine learning, where you learn how machine learning algorithms work. In this course, we will start with a very fast review of basic programming ideas, since you've already done that before. We will learn how to *use* machine learning algorithms to do data

## About the course

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

## About this syllabus

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

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

## About your instructor

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

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

**Important**

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

**Not**

Whet
not m

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

# 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.

Skip to main content

## BrightSpace

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

## Prismia chat

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

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

## Course website

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

## GitHub

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

## Programming Environment

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

### Requirements:

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

### Recommendation:

Skip to main content

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

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

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

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

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

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

Optional:

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

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.

## 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

Python for Data Science is available free online:

## Zoom (backup and office hours only)

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

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

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

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

---

**[1]** Too long; didn't read.

# Data Science Achievements

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

## Learning Outcomes

By the end of the semester

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

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

## Schedule

The course will meet 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.

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

Skip to main content

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

## Achievement Definitions

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

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

Skip to main content

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

## Assignments and Skills

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

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

Skip to main content

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

## Portfolios and Skills

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

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

## Detailed Checklists

### python-level1

*python code that mostly runs, occasional pep8 adherance*

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

Skip to main content

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

### python-level2

*python code that reliably runs, frequent pep8 adherance*

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

### python-level3

*reliable, efficient, pythonic code that consistently adheres to pep8*

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

### process-level1

*Identify basic components of data science*

- [ ] identify component disciplines
- [ ] idenitfy phases

### process-level2

*Describe and define each stage of the data science process*

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

### process-level3

*Compare different ways that data science can facilitate decision making*

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

### access-level1

Skip to main content

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

### access-level2

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

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

### access-level3

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

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

### construct-level1

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

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

### construct-level2

*apply basic merges*

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

### construct-level3

*merge data that is not automatically aligned*

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

### summarize-level1

*Describe the shape and structure of a dataset in basic terms*

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

Skip to main content

### summarize-level2

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

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

### summarize-level3

*Compute and interpret various summary statistics of subsets of data*

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

### visualize-level1

*identify plot types, generate basic plots from pandas*

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

### visualize-level2

*generate multiple plot types with complete labeling with pandas and seaborn*

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

### visualize-level3

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

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

Skip to main content

### prepare-level1

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

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

### prepare-level2

*apply data reshaping, cleaning, and filtering as directed*

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

### prepare-level3

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

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

### evaluate-level1

*Explain and compute basic performance metrics for different data science tasks*

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

### evaluate-level2

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

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

### evaluate-level3

Skip to main content

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

### classification-level1

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

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

### classification-level2

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

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

### classification-level3

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

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

### regression-level1

*identify what data that can be used for regression looks like*

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

### regression-level2

Skip to main content

*fit and interpret linear regression models*

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

### regression-level3

*fit and explain regrularized or nonlinear regression*

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

### clustering-level1

*describe what clustering is*

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

### clustering-level2

*apply basic clustering*

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

### clustering-level3

*apply multiple clustering techniques, and interpret results*

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

### optimize-level1

*Identify when model parameters need to be optimized*

- [ ] identify when parameters might impact model performance

### optimize-level2

*Optimize basic model parameters such as model order*

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

### optimize-level3

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

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

### compare-level1

*Qualitatively compare model classes*

- [ ] compare models within the same task on complexity

### compare-level2

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

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

### compare-level3

*Evaluate tradeoffs between different model comparison types*

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

### representation-level1

*Identify options for representing text and categorical data in many contexts*

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

### representation-level2

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

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

### representation-level3

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

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

### workflow-level1

*Solve well strucutred fully specified problems with a single tool pipeline*

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

### workflow-level2

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

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

### workflow-level3

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

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

# Grading

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

## Principles of Grading

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

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

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

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

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

# How it works

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

These map onto letter grades roughly as follows:

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

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

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

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

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

For each skill these are defined in the Achievement Definition Table

## Participation

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

## Assignments

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

Skip to main content

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

## Portfolio Checks

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

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

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

## TLDR

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

# Detailed mechanics

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

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

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

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

The letter grade can be computed as follows

Skip to main content

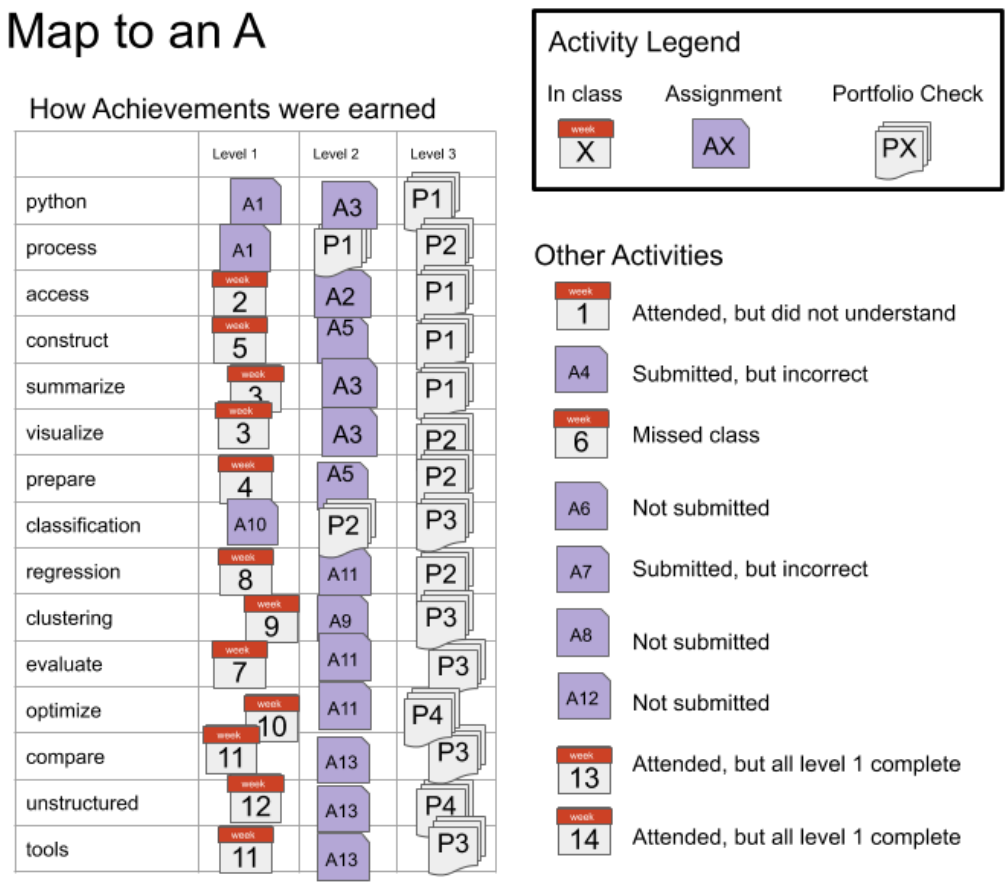
**Important**

this will be revealed after assignment 1

## Grading Examples

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

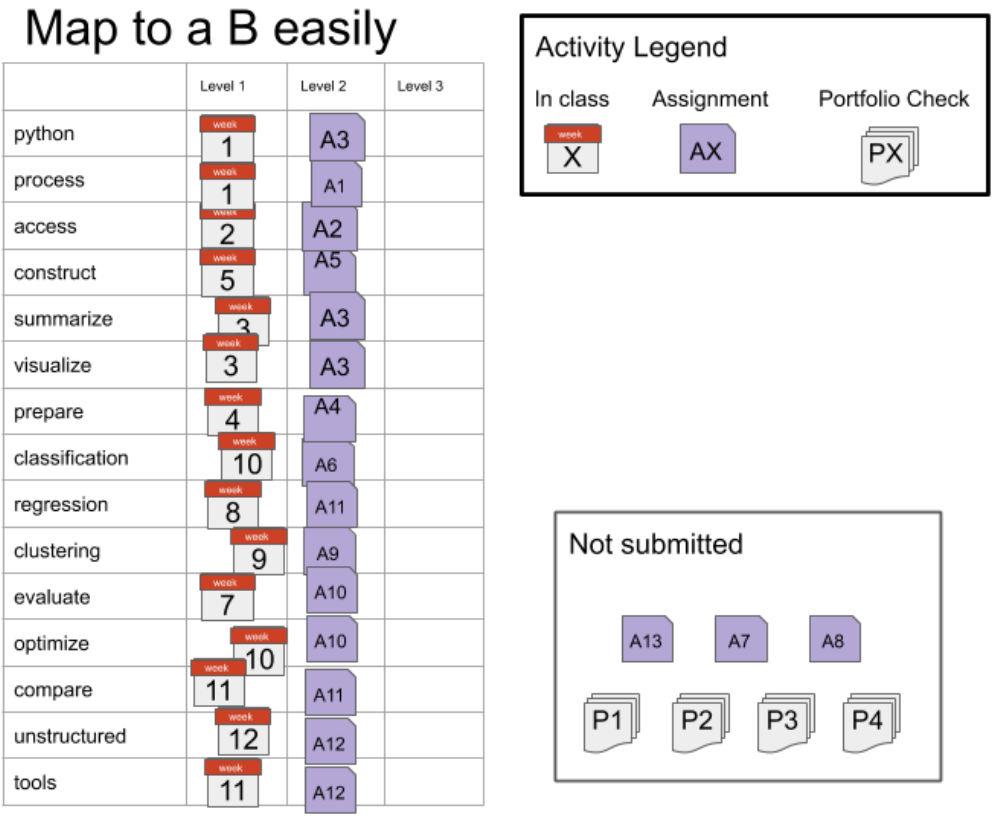
### Getting an A Without Perfection




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

### Getting a B with minimal work

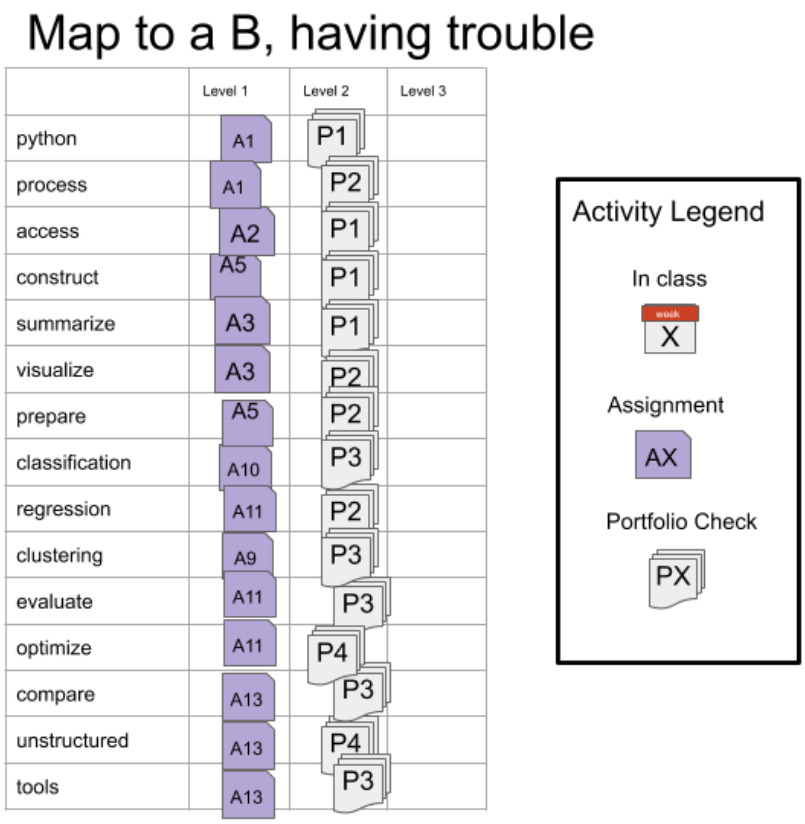
Skip to main content




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

## Getting a B while having trouble

Skip to main content




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

# Grading Policies

## Attendance

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

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

Absences do not require notification.

## Assignment Deadlines and Late Work

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

Skip to main content

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

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

## Portfolio Deadlines and Extensions

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

In this issue, include:

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

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

This request should be no more than 7 sentences.

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

## Academic Dishonesty

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

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

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

For example, if you are violate the academic honesty policy in asignment 4, Prepare level 2 becomes ineligible and you must meet

Skip to main content

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

## Regrading

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

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

# Support

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

## Academic Enhancement Center

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

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

capabilities, to meet a range of accessibility needs. View the synchronous and asynchronous schedules and book online, visit uri.mywconline.com.

# General URI Policies

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

## Anti-Bias Statement:

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

## Mental Health and Wellness

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

## Disability Services for Students Statement:

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

## Academic Honesty

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

- Using material, directly or paraphrasing, from published sources (print or electronic) without appropriate citation
- Claiming disproportionate credit for work not done independently
- Unauthorized possession or access to exams
- Unauthorized communication during exams
- Unauthorized use of another's work or preparing work for another student
- Taking an exam for another student

Skip to main content

- The use of notes or electronic devices to gain an unauthorized advantage during exams
- Fabricating or falsifying facts, data or references directly or indirectly through the use of generative AI
- Facilitating or aiding another's academic dishonesty
- Submitting the same paper for more than one course without prior approval from the instructors

# Communications & Office Hours

> ⚠ **Warning**
>
> Due to Dept Seminar Office hours on 10/13 will be at 5pm instead of 4pm.

## Announcements

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

## Help Hours

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

Zoom links are on the course organization page of GitHub

## To reach out, By usage

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

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

Skip to main content

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

# Tips

## For assignment help

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

## Using issues

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

## For E-email

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

> **Not**
>
> Whet
> not m