CMSC 320: Introduction to Data Science
Final Project: A Tutorial

José Manuel Calderón Trilla

Due by 16:00 (EDT) on May 17, 2021

Motivation

There will be no final exam for CMSC320, instead students are asked to submit a *tutorial* that walks the reader through the Data Science pipeline. The subject matter of this tutorial is far less important that the ability to communicate the approach throughout and a meaningful discussion of the implications/interpretations of the final results.

For the purposes of this tutorial, we will assume that 'The Data Science Pipeline' has the following phases:

1. Data collection/curation + parsing (if necessary)
2. Data management/representation
3. Exploratory data analysis
4. Hypothesis testing and machine learning
5. Communication of insights attained

It is required that each tutorial is a *self-contained* artifact, using a combination of Mardown and Python code within a Jupyter Notebook. This artifact should be *publically available* on the web. Github Pages is a reasonable choice for this, but it is not required (Google Collab is another off-the-shelf solution). We will discuss this later in this document.

The dataset chosen should be publicly available (so that we can replicate your results). Some possible sources of data:

1. A curated list of datasets: `https://github.com/awesomedata/awesome-public-datasets`
2. The U.S. Government is a fantastic source of open data: `https://www.data.gov/`
3. Often individual States will also host their own open data, here is the link for Maryland: `https://opendata.maryland.gov/`

4. Microsoft has a list of open datasets: https://azure.microsoft.com/en-us/services/open-datasets/catalog/

5. The National Institute of Health provides many datasets, here are the COVID related datasets: https://datascience.nih.gov/covid-19-open-access-resources

1 Expectations

In general we would expect a good submission to provide the following at a minimum:

- 1500+ words of prose in English, describing the process throughout and a discussion of the insights attained
- Approximately 150 lines of non-contrived Python
- Well-labelled figures showing important aspects of the analysis
- Links to external documentation and resources that would be useful in understanding the approach.

1.1 Groups

Groups are allowed for the final project, up to 3 people per group. As the group size increases, so would the scale of the expectations: more people should result in a more thorough tutorial. Groups must be pre-registered by April 16th.

EDIT: There have been many questions about what constitutes 'scaling' the expectations. The graders recognize that there are some aspects of the data science pipeline where there may only be one 'right' way to do something. For example, if you're accessing data from a pre-populated SQL database, we do not expect a group to discuss multiple ways of accessing the data! However, there are many aspects in data-visualization, data-exploration, or data-analysis (just as examples) where there *are* multiple methods of accomplishing the same task. We would expect a group to discuss more of those alternatives. Sometimes this results in more code, but often it results in only small amount of extra code, but a significant amount of extra discussion.

2 Examples

The following are links to final projects from past semesters. They should be seen as a rough guide to what is expected and to the variety of topics that can be pursued and not as examples of the highest-scoring submissions.

- https://amulyavelamakanni.github.io/data-science-pipeline-tutorial/: Analysis of Freddie Mac's Single Family Loan-Level data
- https://andrewstehman.github.io/Joe-Flacco-Is-Elite/: Investigation into whether Joe Flacco is an 'elite' quarterback in the NFL
- https://summerzzzy.github.io/: Analysis of global suicide rates
- https://amygracecruz.github.io/: Attempt to predict dementia and Alzheimer's

3 The Deliverable

We recommend 'GitHub Pages' (https://pages.github.com/) for hosting your final tutorial[1] Github provides this service, free of charge, for any public repository hosted on their platform. One of the main benefits is that GitHub does not try to be 'smart' about hosting a Jupyter Notebook, it only hosts a static HTML page. So generating that HTML can be done independently. This also has the side benefit that there is no risk of data sources shifting or changing, causing your project to fail, all data manipulation, analyses, and figures are computed ahead of time and hosted statically.

The following 3 items are required:

1. A GitHub account
2. A repository named `<account-name>.github.io`, where `<account-name>` is your unique GitHub account handle
3. An HTML export of your final tutorial.

You can use the same repository for storing and versioning your final project, in fact I recommend it!), but that is not a require of GitHub pages itself.

There are many tutorials online for using GitHub pages. Personally, I have found the official page to be sufficient: https://pages.github.com/.

3.1 Format of your deliverable

The formatting for the majority of the deliverable is left to your discretion. However, each submission must begin with the title of the tutorial, providing a rough idea of the topic, followed by your name (and all members of the group).

3.2 Submission of your tutorial

The submission to staff is simply the URL for your hosted tutorial. The submission must occur by the University's specified Final Exam time: ***4:00PM EDT on May 17th 2021***. Due to the short time-table for final grades, no late submissions will be accepted.

4 Assessment

The following dimensions of each submission will be given a rating between 1-10:

1. Motivation
2. Understanding
3. Resources

[1] It's where we host the course webpage.

4. Prose
5. Code
6. Communication of Approach
7. Subjective Evaluation

Motivation: each tutorial should be sufficiently motivated. If there is not motivation for the analysis, why would we 'do data science' on this topic?

Understanding: the reader of the tutorial should walk away with *some new understanding* of the topic at hand. If it's not possible for a reader to state 'what they learned' from reading your tutorial, then why do the analysis?

Resources: tutorials should help the reader learn a skill, but they should also provide a launching pad for the reader to further develop that skill. The tutorial should link to additional resources wherever appropriate, so that a well-motivated reader can read further on techniques that have been used in the tutorial.

Prose: it's very easy to write the literal English for what the Python code is doing, but that's not very useful. The prose should *enhance*, the tutorial, adding additional context and insight.

Code: code should be clear and commented. Function definitions should be described and given context/motivation. If the prose helps the reader understand *why* the code should be sufficient for the reader to learn *how*.

Communication of Approach: every technical choice has alternatives, why did you choose the approach taken in the tutorial? A reader should walk away with some idea of what the trade-offs may be.

Subjective Evaluation: does the tutorial seem polished and 'publishable', or haphazard and quickly thrown together? The tutorials should read as well put together and having undergone a few iterations of editing and refinement. This should be the easiest of the dimensions.

4.1 Grades

Once each tutorials has been rated along each dimension, the score for each dimension will be scaled according to the following rubric:

Category	Points Available
Motivation	10
Understanding	10
Resources	10
Prose	20
Code	20
Communication of Approach	20
Subjective Evaluation	10
Total Points:	100