Project: Exploratory Data Munging and Visualization

1 Introduction

No generative AI tools may be used.

It is strongly suggested that you maintain the current version of the milestone in your MS1 Git repository as you work. This will help to prevent any last-minute submitting issues.

The first major step of the project is to perform an exploratory data analysis, where you do data munging, and then generate statistics and visualizations that help to interpret and sanity-check your data. In this assignment, you will conduct the exploratory process described in class.

We will again use peer review in this assignment to improve the quality of your project. As a reviewer, you will provide qualitative feedback on the status of the project. For this milestone, the focus of peer review will be on: the *validation* of data munging processes, and the *reproducibility* of the process used to generate processed (cleaned/transformed) data and visuals.

Thinking ahead: later in the semester you will be writing a paper that presents your data science project. The analysis you perform here and the visuals you create can make up a substantial part of that paper. Also: it would be ideal if your data munging set the scene for the future use of a classification or regression algorithm.

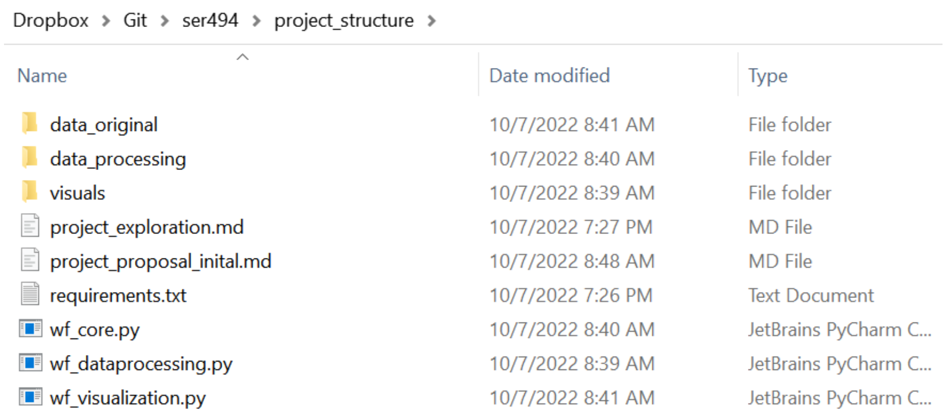
1.1 Project Structure

Projects that do not follow the program and file structure described here, may not be worth credit. Your peer reviewers and the grader will be expecting this format and it is essential in making sure they can quickly find what they need.

In this part of the project, you will develop a workflow that analyzes your data. It must be structured as follows:

Source Code and Documentation: create the following five or six files in the root of your repository:

- (if not using existing data files) wf_datagen.py: This will contain the code for fetching (scrapping) or generating (simulating) the data for your project. Executing it will save files into a folder called data_original.
- wf_dataprocessing.py: This will contain the code associated with data munging. Executing it will generate a set of processed data files.
- wf_visualization.py: This will contain the code associated with basic statistics, correlations, and visualization. Executing it will generate all of the visuals for your data, provided processed data files already exist.
- wf_core.py: This will call wf_dataprocessing to process data followed by wf_visualization to generate the statistics and visuals. It should be relatively simple, it serves only to orchestrate the other files.
- project_exploration.md: This will contain documentation for your work on this milestone as outlined later.
- requirements.txt: Contains a list of all project dependencies. To be discussed in subsection 2.1.1.



Name	Date modified	Type
data_original	10/7/2022 8:41 AM	File folder
data_processing	10/7/2022 8:40 AM	File folder
visuals	10/7/2022 8:39 AM	File folder
project_exploration.md	10/7/2022 7:27 PM	MD File
project_proposal_inital.md	10/7/2022 8:48 AM	MD File
requirements.txt	10/7/2022 7:26 PM	Text Document
wf_core.py	10/7/2022 8:40 AM	JetBrains PyCharm C...
wf_dataprocessing.py	10/7/2022 8:39 AM	JetBrains PyCharm C...
wf_visualization.py	10/7/2022 8:41 AM	JetBrains PyCharm C...

Figure 1: Basic project file structure. Your project will contain all of these files and folders, and potentially a few more. project_proposal_initial.md will remain unchanged from MS1.

If you wish to further subdivide your .py work into additional files, use the above names as prefixes. For example: you would still have a wf_dataprocessing.py file that processes your data, but it might call methods from wf_dataprocessing_fillblanks.py.

Data: all of the original data must be stored in a sub-folder called "data_original". The processed data must be stored in a sub-folder called "data_processed". If you are using an online source for your data, any files you use will be saved in data_original. If you are scrapping a website (or similar) you should save the collected data in data_original before applying munging transformations.

Visuals: all of your generated visuals/charts/figures/etc must be stored in a sub-folder called "visuals".

An example of the basic structure for your project is shown in Figure 1. The overall design for the completed workflow is shown in Figure 2.

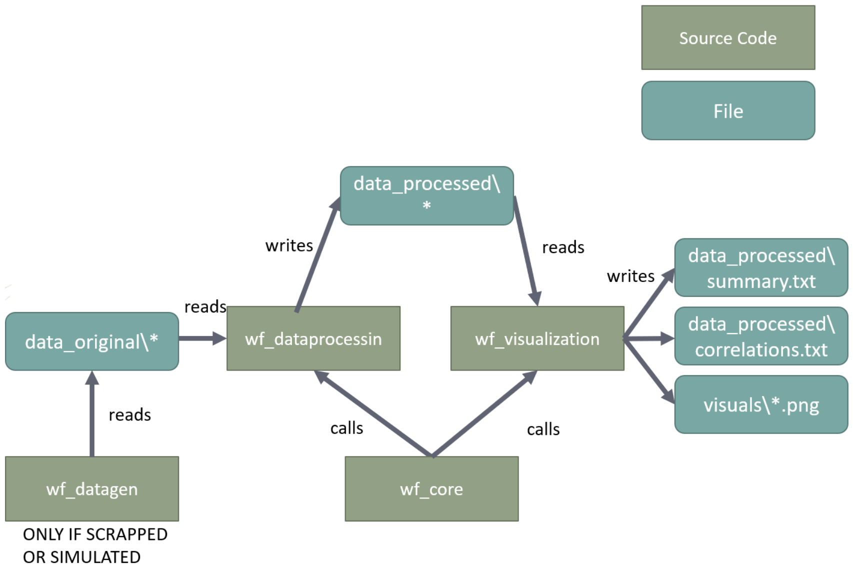


Figure 2: High-level workflow design.

2 Requirements

For this assignment, you will perform exploratory data analysis. All data munging and visualization must be implemented in Python 3.9 or newer. The original data files that you process should be placed in a sub-folder called data_original. If you are scrapping (or simulating) data, the file wf_datagen.py will be saved into the data_original folder. Once the data is available, the following exploratory steps must be followed:

1. Answer basic questions:

   - What: identify the author(s) of the dataset(s), when the data set was constructed, how many records it contains, the meanings of the fields/attributes, and compute an MD5 hash for each file. (If a file was downloaded, provide the URL and the MD5 of that file. **The MD5 hash of the download and the one in your repo must match.** If the file was generated (e.g., scrapping, simulation), include the URL source of the data or tool.)
   - Where: project_exploration.md
   - **Graduate students only:**
     - See MS1 requirements on dataset age and source.
     - If you find that your dataset is synthetic, please consider carefully if it is possible to generate new knowledge with it (meaning nobody has seen the results before). Generally, the answer will be no (if it's synthetic then someone knows what is happening since they created the rules by which it was generated), and if you are unable to do so, you may need to repeat much of the work in this milestone later on in the course.

2. Look for interpretable records:

   - What: select two (2) records from your data. This might be a row in tabular data, or a node in graphical data. Explain each of them in plain English, and discuss if it is "reasonable" for the meaning of the data.
   - Where: project_exploration.md

Data munging happens here! Notice that it is not explicitly called out in the list of exploratory steps. Your goal is to do whatever data munging is required so that you can perform the following steps that analyze the data. Your data munging code will be put in wf_dataprocessing.py. You must generate a series of files (in a data_processed sub-folder) which can later be loaded by your visualization code.

3. Compute summary statistics:

   - What: Select five features in your data, at least three quantitative ones and one qualitative. For quantitative features compute the min, max, and median. For qualitative features compute the number of categories, most frequent category (multiple if needed), and least frequent (multiple if needed). See slides for a sample tabular output.
   - Where: code will go in wf_visualization.py, output should be saved to "data_processed\summary.txt".

One important purpose of computing summary statistics (and the remaining steps) is to provide a characterization of the distribution of the data you're analyzing. Eventually, you will need to think of this as creating the "evidence" that the distribution of your dataset is representative of the real world, and hence, the results of your work are transferable to the real world. This is in contrast to a dataset that is not representative, often due to biases in the data or from having a dataset that is not large enough to capture significant variation within your domain.

For some projects (say: analyzing student grades), the above may be as simple as showing the dataset has both bad students (min) and good students (max), with a reasonable number between them (median). This can even be checked by plotting the data and looking for something like a normal distribution.

Warning: projects using data of a more "compound" nature may extra effort here. Suppose that you are working on a project to apply machine learning to clean up data in astronomical images that pictures of the starfield which contain noise from passing satellites. In this case, the data is images. Again, let's suppose that we have some huge ZIP file of images. These images form a distribution but it is hard to characterize. It doesn't make sense to do something like extract the height/width/etc of the image files and treat that as a quantitative feature. That information is not useful to the problem (by resizing, we could take the same image and make it "different"... size isn't a useful metric here!). Instead, we would want to transform our dataset into quantitative features. For example: the number of stars, number of streaks, number of occluded stars, etc. These are now useful quantitative features that can be analyzed. It can be time-intensive to get these features, often requiring manual annotation or a simulation backend. These features cannot be extracted with ML approaches since those methods are approximate and we require a ground truth.

More data munging may happen here! This is where you may start to see outliers, which may be removed with justification.

4. Compute pairwise correlations:
   - What: For the quantitative features you selected, create a matrix of correlation coefficients between all columns. See slides for a sample tabular output (notice that only half the matrix needs to be created as it contains redundant information).
   - Where: code will go in wf_visualization.py, output should be saved to "data_processed\correlations.txt".
5. Class breakdowns
   - This is one of the six steps but we'll skip it due to time constraints.
6. Plots of distributions
   - What: Create scatter plots for all pairs of quantitative features that you selected. For example, if you have quantitative features A, B, and C, you will construct AB, AC, and BC. For qualitative features, construct a histogram. For example, if have qualitative features D and E, you will construct D and E. All images must be in .png format and follow guidelines for quality visuals from the lecture.
   - Where: code will go in wf_visualization.py, output should be saved to "visuals*.png" where * represents different filenames of your choice.
   - **Note: Do not accidentally commit solutions to the visualization homework in your repo.**

The program (workflow) that you construct for generating the statistics and visuals must be designed in a way that others can execute it to generate the results. Make sure that paths are not hard coded and that you are running the most recent version of any packages used. In addition, you will need to fill out the file project_exploration.md (see Canvas for a template file, remove "_template" from the name) with information on your exploratory process:

1. Title: create a concise title for your work.
2. Author: indicate your name.
3. Date: indicate the date of submission.
4. Basic Questions and Interpretable Records: *will already have been completing when conducting the process.*
5. Background Domain Knowledge: Write a basic introduction to the domain of your project. It must between 200 and 500 words. Protip: doing well on this description can help your peer reviewers to understand the content of your work and thus award more points.
   - Integrate and cite at least three (3) sources of information. Any accurate source may be used (e.g., not just textbooks but blogs by reputable authors too). This section should not include technology discussion, your focus is on the topic. For example: if you have a project on evaluating how to determine if users liked or disliked using a program, you would not discuss how classification works. Instead discuss the important elements of UX design and how different designs are evaluated in industry. (The sources from your project proposal may be used here if they are about the domain and not technical.)
   - Any citation format that can easily help the reader to retrieve the work is permissible.
6. Data Transformation: for any transformation you apply (e.g., smoothing, filtering, imputation, standardization, etc.):
   - Describe the operation that you are applying.
   - Justify that the operation does not change the semantics of the data, discard usable data, introduce errors, or introduce outliers.
7. Visualizations: For five (5) of the visualizations you produced, write a short (two to three sentence) but thoughtful analysis of the relationship revealed in the data being plotted.

2.1 Allowed Packages

For this milestone, the following packages are allowed:

- Anything available by default on a Python 3.x install.
- Matplotlib (matplotlib)
- Numpy (numpy)
- Pandas (pandas)
- Requests (requests)
- BeautifulSoup4 (beautifulsoup4)
- NLTK (nltk)
- CV2 (cv2)
- Seaborn (seaborn; this approval does not extend to scipy)

We may allow use of other packages on a case by case basis. Please check with the instructor or TA at least 72 hours prior to the deadline.

```
beautifulsoup4==4.11.1
nltk==3.7
numpy==1.19.1
pandas==1.5.0
PyYAML==6.0
requests==2.24.0
```

Figure 3: Sample requirements.txt generated by pipreqs for HW02.

2.1.1 requirements.txt

Once you have finished your programming work, you should generate requirements.txt. This is a textfile that explicitly lists all of the dependencies for a Python project. We suggest using the package pipreqs to generate this file but other tools are fine as well. To use pipreqs, you'll use "pip install pipreqs" to install it, and then execute "pipreqs" from the root folder of your project. See Figure 3 for a sample.

3 Evaluation Process

After submitting, we will conduct a single-pass peer review. **Since peer review requires running another student's project, please be careful. The easiest approach would be to use a VM, although reading the code before running it is often enough.** Reviewers are expected to complete the rubric associated with the assignment (see Appendix) and to generate qualitative comments on the work and justify the ratings. **You must provide a justification for each criterion.** Only selecting a level is not sufficient to earn full credit for your peer reviews. Your selections must also be accurate.

- For this assignment, we will use the peer review functionality provided by Canvas.
- Peer reviews: you will have two or three non-blinded reviewers. The staff may review as well.

A few notes:

- It appears that Canvas only lets you submit the rubric once - be careful.
- Make sure that you use the rubric to store your main scores and comments. If needed a file can also be attached but the Canvas rubric needs to be filled out to track who has completed their peer reviews.

4 Deliverables

For this assignment, you will receive two separate grades, which evaluate:

- Project: Data Munging and Visualization (Canvas, GitHub): see Section 2 for details. [40 points]
  - The results of your analysis (not just project_exploration.md but the files containing the processed data, statistics, and the visuals) must be provided.
  - Your work should be: 1) uploaded to your Git repository, and 2) uploaded to Canvas (compress and upload your repository). Both are due by the deadline and must be identical.
- Project: Data Munging and Visualization Peer Reviews (Canvas): how well you conduct the peer reviews. Grading will be based on accuracy and providing useful feedback on the project. [40 points]

5 Next Steps

See Canvas for the following two assignments:

1. MS4: Data Munging and Visualization Peer Review
2. MS4: Revised Project Proposal and Questions

Appendix A: Peer Review Rubric (tentative)

Criteria	**Proficient**	**Competent**	**Novice**
Basic Questions	Correctly identified dataset author, when data set was constructed, the number of records, meaning of fields, and computed MD5 hashes.	Correctly provided some but not all: identified dataset author, when data set was constructed, the number of records, meaning of fields, and computed MD5 hashes.	Provided inaccurate information on: dataset author, when data set was constructed, or MD5 hashes.
Interpretable Records	Explained two records in plain English, and discussed if they are reasonable for the meaning of the data.	Explained two records in plain English, but explanation not sound or did not discussed if they are reasonable.	Explained less than two records in plain English.
Summary Statistics	Properly computed (and saved) summary statistics for five features.	Properly computed summary statistics for some but not all of the five features.	Incorrectly computed summary statistics.
Pairwise Correlations	Properly computed (and saved) a matrix of correlation coefficients between all quantitative features selected.	Properly computed a matrix of correlation coefficients for some but not all quantitative features selected.	Incorrectly computed matrix of correlation coefficients.
Plots of Distributions	Created quality scatter plots for all pairs of quantitative features selected, and histograms for qualitative features.	Created plots for all selected features, but either used wrong plot type or incorrect data.	Did not create any plots, or all plots had significant quality issues.
Data Source Provenance	Hash for all data files from online sources match hash of file in repository.	Hashes for online data files did not match hashes of files in repository.	Author did not include hashes for all online data sources.
Data Transformation	Accurately described all transformation operations applied, and transformations were sound for the semantics of data.	Accurately described all transformation operations applied, but transformations were NOT sound for the semantics of data.	Inaccurately described some transformation operations applied.
Visualizations Description	For each visualizations, wrote a short but thoughtful analysis of the relationship revealed.	For each visualizations, wrote a (poor) analysis of the relationship revealed.	Did not describe all visualizations or give a misleading description.
Reproducibility	Workflow was set up such that running it generated virtually identical outputs as the original author obtained.	Workflow generated different files, to the extent that means of statistics or visuals changed.	Workflow indicated multiple aspects that were hard-coded to author's system.

Notes for reviewers:

- If someone fails to structure their project as outlined in this document, then that is grounds for you to make a deduction if you are unable to find what you need to evaluate.
- Reviewers should make sure they have Python 3.9 or newer installed, as well as all packages mentioned in requirements.txt, prior to attempting to reproduce outputs.