IE 555 – Programming for Analytics

Homework #4 – Data Visualization

Due Date: Monday, November 6, at 3:00pm (in class)

The goals of this homework assignment are fourfold:

1. To solidify your understanding of plot types that we covered in class, by reproducing the format of a plot according to some specifications.
2. To give you practice generating new types of data visualizations, beyond what we covered in class. We only saw a handful of fairly simple examples.
3. To give you practice with independent learning. The vast majority of the code you will write in the future is not found in a single textbook or a single Website; you'll have to aggregate bits of information from multiple sources (including your own Python knowledge, of course).
4. To give you practice writing documentation that will enable others to leverage your work. It is important that others can make use of what you created. This requires good documentation.

This is an INDIVIDUAL assignment.

Assignment Details

This assignment is divided into two (2) parts. **It is important that you follow these instructions exactly.**

Part 1 - Generate Plots According to Specifications

There are three (3) problems for Part 1:

1. **A Scatter Plot with a Line.** For this problem, suppose that you are given measurements taken along the length of a very rough part. The x-coordinate represents the distance from one end of the part; the y-coordinate represents the value of the measurement.

   (a) Your Jupyter notebook should read from a file named `scatter_data.csv`, which will be saved in the same directory/folder as your `.ipynb` notebook.
   - A sample/practice file has been provided for you. However, the TA will replace this file with different data.
   - Your code should ignore any rows that begin with the percent sign (`%`), which denotes a comment.
   - The data file will contain two columns, representing an x-coordinate and corresponding y-coordinate value.

   (b) Plot the data points as **green triangles**.

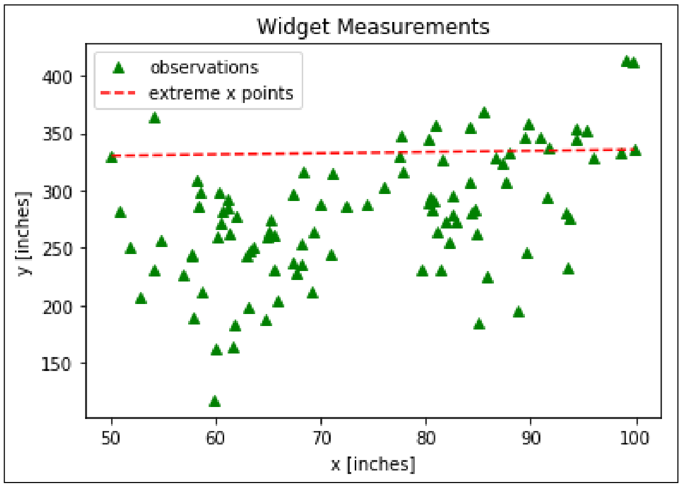
   (c) Plot a line from the left-most point (minimum x-value) to the right-most point (maximum x-value). The line should be **dashed** and colored **red**. *Note: There's really no practical reason for drawing such a line; this is just to practice your plotting skills.*

   (d) Include a title for your plot.

   (e) Label the x- and y-axes.

   (f) Include a legend.

   Using the sample data provided, your plot should match what is shown here:



2. **A Histogram.** In this problem, you'll create a histogram to help visualize the distribution of letter grades for a class.

   (a) Your Jupyter notebook should read from a file named `student_grades.csv`, which will be saved in the same directory/folder as your `.ipynb` notebook.
   - A sample/practice file has been provided for you. However, the TA will replace this file with different data.
   - Your code should ignore any rows that begin with the percent sign (`%`), which denotes a comment.
   - The data file will contain two columns. The first column of each row will contain a student ID; the second column will contain each student's average score.

   (b) You are asked to create a histogram of student scores, as grouped according to the following bins:

Grade	**Score Range**
A	[90, 100]
B	[80, 90)
C	[70, 80)
D	[60, 70)
F	< 60

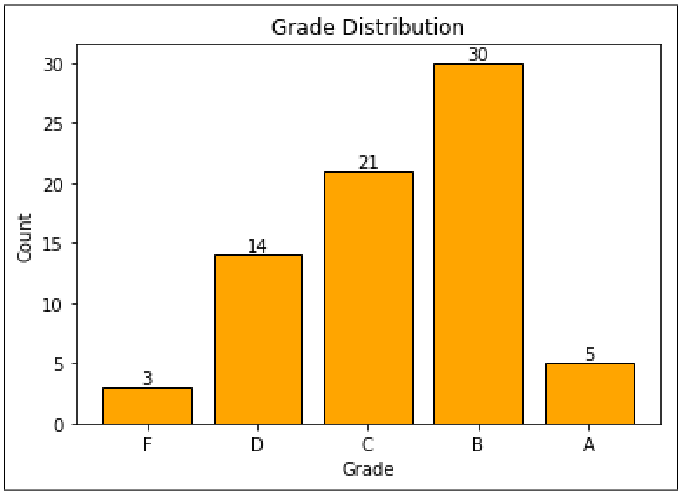
   (c) Each bar should be colored **orange**, with a thin **black** outline.

   (d) The counts for each category/bin should be displayed above each bar.

   (e) Appropriately Label the x- and y-axes.

   (f) Be sure to include a descriptive title for your histogram.

Using the sample data provided, your plot should match what is shown here:



3. **Barplots and Boxplots in the Same Figure.** In Operations Research we often deal with problems for which it is very time-consuming to find the/an optimal solution. In such cases, heuristics are useful for finding what are (hopefully) "very good" solutions. You will be given data describing objective function values for numerous maximization optimization problems. Each problem has an ID number (to uniquely identify the problem). Each problem was then solved optimally, and was also tested with three (3) different heuristics. We want to visualize the "optimality gap" for each heuristic. That is, the percentage difference between the optimal objective function value (OFV) and each heuristic's OFV.

   (a) Your Jupyter notebook should read from a file named `solution_data.csv`, which will be saved in the same directory/folder as your `.ipynb` notebook.

   - A sample/practice file has been provided for you. However, the TA will replace this file with different data.
   - Your code should ignore any rows that begin with the percent sign (`%`), which denotes a comment.
   - The data file will contain three columns. The first column of each row will contain a problem ID; the second column will describe the solution approach used to solve the problem; the third column will contain the corresponding OFV.
   - You may assume that the second column will only contain the following text strings: "optimal", "genetic algorithm", "simulated annealing", and "tabu search".

   (b) Create a single figure with two subplots. The figure should be labeled "Comparison of Optimality Gaps for Heuristics".

   i. The first (left) subplot will be a **barplot** showing the average optimality gap for each heuristic. Each bar should be colored orange, with thin black outlines; label each bar along the x-axis; include a label for the y-axis, and provide a title for the barplot.

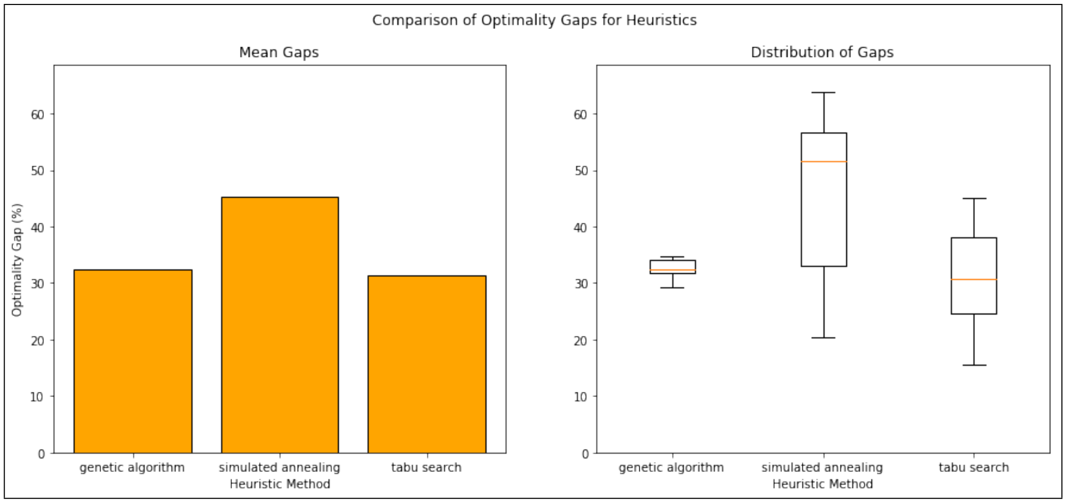
   ii. The second (right) subplot will be a **boxplot** showing the spread of optimality gaps for each heuristic. Label each bar along the x-axis; color the median line orange; be sure that the scale of the y-axis is the same for both plots; and provide a title for the boxplot.

   NOTE: The optimality gap for a heuristic, represented as a percentage, is calculated as:

$$\text{gap} = \frac{(\text{optimal OFV}) - (\text{heuristic OFV})}{(\text{optimal OFV})} * 100,$$

   where positive gaps indicate that the heuristic's solution was suboptimal for a maximization objective.

Using the sample data provided, your plot should match what is shown here:



Part 2 - Explore New Plot Types

In this part you are asked to choose one (1) different plot type from the following sources:

- Matplotlib Gallery – https://matplotlib.org/gallery/
- Matplotlib Tutorials – https://matplotlib.org/tutorials/
- Seaborn Examples – https://seaborn.pydata.org/examples/index.html

1. For your plot, find an appropriate source of data online.
   - You must provide links to the plot type.
   - You must provide links to the online data source.
2. Create a plot using your data source. Since you're using a unique data source, your plot should look different from what you found in the galleries/examples.
3. Provide an explanation of your code.

There are a huge number of plot types, and a significantly larger number of data sources online. Therefore, the work you submit should be clearly distinct from all of your classmates. **Also, your plot should be distinct from the bar/scatter/line/boxplot figures we created in class. Create something new/exciting/interesting.**

Submitting Your Assignment

You will upload your assignment to GitHub in a repository that the TA will create for you. A template Jupyter notebook, named `sample_hw4.ipynb` will be pre-loaded in your GitHub repository.

Your Jupyter Notebook should be named `ubusername_hw4.ipynb`, where `ubusername` should be replaced with your UB username. For example, my UB username is `cmurray3`; my notebook will be named `cmurray3_hw4.ipynb`.

There is no auto-grader for this assignment. Instead, we will be grading the assignments **in-class** on Monday, November 6 and Wednesday, November 8. It is important for you to attend class on those days; otherwise, you will not receive credit for this assignment.