Final Project Description

Due Date : Dec. 15, 11:59PM
Assessment Value: 125 pt.

In this course, we have explored eight foundational algorithms in data science through a series of labs. We have also learned key principles in numerical algorithm design, such as computational efficiency, conditioning, and stability. These components contribute to one of the major goals of this course, to provide students with building blocks to teach themselves the new algorithms that they will encounter in their profession lives.

The goal of our final project is to put this plan into practice. Each student will select and implement an algorithm that was not covered in class. The student should then apply their implementation to solve a problem using a real or simulated dataset. As evidence of their work, the student will need to provide both the Python code for their implementation and a brief write up. Collectively, these two files should cover five components, which are comparable to the sections of our labs:

1. Background
2. Design
3. Implementation
4. Validation
5. Reflection

In addition, please email or discuss with the instructor what algorithm and problem you intend to use. This will verify the suitability and feasibility of your project. The contents of these sections, as well as their point values, are detailed below.

Background (Addressed in Write-Up; +20 pt.)
The Background section should be a written section with three components:

1. *Goals (+5 pt.)*
   You should explain what procedure or method you wish to accomplish with your algorithm. This explanation should include both what the algorithm fundamentally does and what task you will complete on your dataset using your algorithm.

   Using an example from class, if a student were writing about using a QR decomposition to solve a least-squares problem, the explanation of what the QR decomposition algorithm does would state that a QR decomposition expresses a matrix as the product of an orthogonal matrix and an upper triangular matrix. The explanation of the task could be to fit a multiple regression model to predict one variable in their dataset using the other variables.

2. *Background* (*+10 pt.)*
   You should provide a basic explanation of the mathematics behind your algorithm and/or how your algorithm relates to one of the algorithms that we discussed in class. Extending the previous example, if a student were writing about using a QR decomposition to solve a least-squares problem, the student could first explain why multiplication with the orthogonal Q^{T} matrix does not change the least-squares problem's solution and comment on the back-substitution algorithm's reuse on the R matrix to find the solution.

3. *Description of Your Dataset (+10 pt.)*
   The format for this component will depend on whether a real or simulated dataset is used. For a real dataset, the description should contain:
   a. A citation for the original source
   b. The sample size
   c. All relevant variables
   d. A basic description of the population (*e.g.*, when the data was collected, where the data was collected, what were the objects comprising the population)

   For a simulated dataset, the description should contain:
   a. What functions you used to simulate your dataset
   b. The parameters used to simulate the data

   Alternatively, you may provide a script that creates your simulated dataset and reference the script in your written description. *Regardless of how this information is provided, though, be sure to set the random seed before simulating your data so that you can obtain the same dataset every time your code runs.*

Please be sure to cite any sources that you use. Your background information does not need to be highly detailed. One to two pages (double-spaced, 12 pt. font) should be a sufficient length.

Design (Addressed in Write-Up; +20 pt.)
Write out all necessary algorithms in pseudocode. The exact format is not pivotal, but your pseudocode algorithm should have the following features:

1. Clearly state the inputs into your algorithm
2. Clearly state the outputs from your algorithm
3. Clearly indicate what lines are within loops/control statements

You can use the pseudocode algorithms in the labs as an example.

Tip: Complete your pseudocode algorithm *before* you begin to write your implementation. The goal of writing pseudocode is to have a blueprint in place before tackling a true implementation.

Implementation (Addressed in Python Script; +30 pt.)
In this component, you should implement the algorithms necessary to accomplish your stated goals. As a result, this component should be addressed within a Python script. Your work will be graded based on two components:

1. *Documentation (+5 pt.)*
   Each function should have a basic description that includes:

   a. The purpose or goal of the function
   b. What each function argument represents and the type/data structure that the argument should be provided as
   c. What each output from the function represents and its type/data structure

   You can use the documentation for the various labs' helper functions as a guide.

2. *Correctness (+25 pt.)*
   Whether your implementations provide correct outputs for a set of arbitrarily chosen inputs, assuming the reader follows all the directions provided in the documentation.

Validation (Addressed in Both Write-Up and Python script; +40 pt.)
Similar to the Validate sections in our labs, the validation component should have two subcomponents:

1. *Test Case (+20 pt.)*
   You should design a simple test case that you can use to determine whether your algorithm implementation(s) are working properly. Then, you should work out using pen and paper what outcome your algorithms should produce if they are in fact working properly. Please provide the pen-and-paper solution in your write-up and the code for to carryout your test in your Python script.

2. *Accomplish Stated Goals (+20 pt.)*
   Within the same Python script where you implemented your algorithms, you should also provide code to accomplish whatever your stated task was in the Background section.

Reflection (Addressed in Write-Up; +10 pt.)
There is always room for improvement. In this final component, describe how you could improve your work in the future. These improvements could involve (but are not limited to):

- Incorporating more computationally efficient method(s) into the process
- Capitalizing on different tools (*e.g.*, general-purpose computing on GPUs, cloud computing, *etc*.) to obtain lower wall times
- Extending your work into a new algorithm,

Be specific. You should provide exact and feasible improvements, including justifying why this improvement would be feasible. *If you use additional sources, please be sure to cite them.*

Once you have completed your write-up(s) and Python script(s), submit both your PDF and Python script using the following algorithm:

1. Go to our class Blackboard page, and select Modules from the left panel.
2. Click Final Project.
3. Click on "Final Project." This title is also a link to the submission portal.

4. Scroll down to the Assignment Submission section. Click Browse My Computer, locate your files in the pop-up window, and click Open.
5. Click Submit.