Final Project Description

----------<Important Dates>----------

Checkpoints: Feb 27th, March 10th, March 27th , April 7th , April 21st

Partner Survey: Feb 27th

Final Due: April 24th

----------<Repo>----------

Github repo: https://github.com/jdb051/csci204Project

Starter code will be posted on the repo. You may download and compare your code to this starter code. Some code already exists. Some will be added as needed to the repo to aid in each checkpoint. You may choose to use a repo too if you would like for the project, but not required. It is best to check if I have updated or added new code to the repo at least on a weekly basis. This can be done by clicking commits in the upper left-hand corner on the webpage.

CSCI204 will contain one final project. This final project will be around the concept of text analysis for the Enron Email Set. Text analysis is a wide area that includes data mining. Text analysis is a huge business and used by companies such as eBay, Facebook, Microsoft, and Google. The Enron dataset is very interesting because of how it has been used in the past to see into the fall of Enron. More can be read about Enron here: https://en.wikipedia.org/wiki/Enron

In particular, this project will focus on four parts: good programming, data organization, data parsing, and data analysis. To insure that the final project goes smoothly for students, the final project will be broken into five checkpoints, a final turn in, and a presentation. The five checkpoints will account for 33% of the project grade.

----------<Partner>----------

This project requires you to work with a partner. You will have to fill out a survey on Moodle to identify your partner by Feb 27. Only one person needs to fill out the survey. If you have a valid reason why you cannot work with a partner, you must request a waiver for this from me by Feb 21st via email.

----------<Check Points>----------

Checkpoints are due the days listed above. You will be required to submit these checkpoints in either two ways. The first way is via git repo link. The second is via a zipped file (.zip) with your name and

checkpoint number (e.g., studentname2.zip) on Moodle. You must zip the whole file structure (overview, src, eval, … ,etc) into one file. This must include all the python files used in your program (not just the ones modified by that checkpoint).

These checkpoints will be graded in the following manner. Checkpoint will be graded out of 2 points. First, the project will be attempted to run. Second, sample inputs will be tried from the menu. If any these input fail, this counts as missing one step of the checkpoint. Lastly, the code requirement of each step will be hand checked. If any parts of these requirements are not fulfilled, this will count as a missed step. Based on the number of steps completed, a score will be assigned to the checkpoint. A grade of 2 will be assigned if the student fulfills over 85% of the requirements. A grade of 1 will be assigned if less than 85% percent but more than 50% is fulfilled. A grade of 0 will be assigned if less than 50% is fulfilled. These can be considered A, C, and F. All checkpoints will account for a total of 33% of the total final project grade.

Below is an outline of what the checkpoints will cover. **This is only an outline, and you should read each checkpoint for the details!**

----------<Code Comments>----------

Though code comments are not checked in the checkpoints, code comments will be checked in the final project. They will account for 10% of the final project grade. The following will be checked for comments.

1. At the beginning of each py file, include your name and a brief overview of what is done in the file using a docstring.
2. Before every function, you must state the purpose of the function and any assumptions the function makes (A good way to outline these assumptions is with pre/post conditions).

----------<Final Project Write-up>----------

The final project requires a write-up that will be turned in with the project. This write-up will have two sections. Section 1 will outline a brief user manual of your project. This should give general directions to the user to clarify how to use the program. Section 2 will be a reflective piece about your own programming experience. The reflection should outline what you learned, what you struggled with, and how you overcame these troubles. The length of the write-up should be between 2-4 pages doubled spaced. 20% correct spelling and punctuation. 80% fulfill both sections.

----------<Correct File Management>----------

10% of the final grade will be based on correct file management. <u>If no file management is used (all in one file), the instructor holds the right to give the student a 50% on the whole project</u>. The key ideas examined will be the use of breaking down the code into pieces that make sense and make the code easy to read.

----------<Correct Class, Function, and Variable Use>----------

This section will grade multiple ideas. This includes but not limited to correct variable names (ones outlined in checkpoints), correct variable style (e.g., class starting with capital, simple variable starting with lower case letter, full words and no single character names for key variables). This includes the correct use of private variables, setters, and getters for classes.

----------<Correct Execution>----------

Correct execution includes multiple topics. First and foremost, correct execution includes that your code will run with Python version 3.5+ and will depend on a number of libraries such as numpy, matplotlib, scikit, and more. It is strongly recommended for users to install Anaconda (https://www.continuum.io/downloads) as it contains all needed packages and runtimes. If you do not know that your code will run under these standards, please see the instructor during office hours. The second topic will be that all functions execute as they should as outlined in guidelines. Though super-efficient code may not be needed except where an algorithm is outlined, all code must be able to run on my system in a "reasonable" amount of time.

----------<Point Breakdown>----------

Area	Description	Percent
Checkpoints	Checkpoints are graded in three categories (2,1,0). See Checkpoint section for more details.	33
Code Comments	Code comments should be constructive. See comment section.	7
Final Project Write-up	One document outlining how to use your program and what you have learned.	8
Correct File Management	Each class has py file.	7
Correct Class, Function, and Variable Use		15
Correct Execution	Must run and work as outlined.	30
	Total:	100

----------<Final Project Presentation>----------

Each group of partners is required to make a short 5-minute presentation. This will account for 1 checkpoint. The goal of this is for you to talk about what you learned, liked, disliked, or wanted to add to your project. A time schedule will be setup for this the week before the presentations.

----------<Extra Credit>----------

Extra credit will be given for additional improvements to the project. It is recommended that you ask about these to the instructor before you implement them.

----------<Checkpoint 1 Feb 27th>----------

Getting started. You will have to download and run the Python files given. This will help test if you have the right libraries (packages) installed on your system. You will also need to work on the DocumentReader to read a MIME file given in the train and eval folders. More information about this format can be found in the details of checkpoint 1.

----------<Checkpoint 2 March 10th>----------

You will now have to add an interface that will load all training and eval documents with good exception handling. You will have to add code to the stat.py file for findFreqDic, topNSort, and bottomNSort using the built in sort of Python list. You should add the needed code to plot.py for twoDScatter and twoDBar. You should add the interface for Topic Analysis of Train and Topic Analysis of Eval. See checkpoint 2 write-up for more details and additional requirements.

----------<Checkpoint 3 March 27th>----------

In this checkpoint, you will make more steps to clean up data and implement data structures. This includes the data structures we study in class, as well as, a text filters and the required interface. See checkpoint 3 write-up for more details and additional requirements.

----------<Checkpoint 4 April 7th>----------

This is a bigger checkpoint. You will need to add our decision algorithm (our first decision method), sktree, and additional requirements for topNHeap and bottomNHeap in stat.py. See checkpoint 4 write-up for more details and additional requirements.

----------<Checkpoint 5 April 21st>----------

In this checkpoint, you will add skPCA and our sorting algorithms to help speedup our execution. By the end of this checkpoint the total project source code should be 99% done. See checkpoint 5 write-up for more details and additional requirements.