CSC 202: Intro to Machine Intelligence
Natural Language Processing Project

In our unit involving Natural Language Processing, we covered how to analyze text in Python using the *TextBlob* and *Spacy* modules. In this project, you will use natural language processing for text-mining to analyze one or more texts. **Be creative!**

Analyze one or more sets of texts by completing the requirements below. Example project ideas include

- Analyzing a text, such as an e-book, Wikipedia entry, news article, political speech, or song lyrics
- Comparing texts (such as different Wikipedia pages, different e-books, different chapters from the same book, different songs, etc)

The following resources may be useful:

Song lyrics: https://www.lyrics.com/

E-books: Project Gutenberg

Movie scripts :

- The Internet Movie Script Database
- Awesome Film
- Screenplays for You
- Movie Scripts and Screenplays

You must turn in a Jupyter Notebook that carries out the appropriate analysis. Your notebook should begin with your name and a one paragraph summary describing the main goals of your project (e.g., to find the most common words in an e-book). Each code cell should be preceded by a markdown cell that describes what the code does (as in the class notes). Your Notebook should be correctly formatted with an appropriate title, section headers, and subsection headers. [20 points]

1. Complete **one** of the following to collect data to analyze [25 points]:

- Retrieve data from at least one Wikipedia page (see Lab 8 for an example)

- Read data from a plain text file, which can be done using the code below (the file should be a plain text file from the same directory as your Notebook):

-
  ```
  f = open('file.txt', 'r')  # open a file for reading
  text = f.read() # stores the text of the file in string text
  f.close() # close the file
  ```

2. Complete any **three** of the following [25 points each]

   a. Generate a table showing the 20 most frequently used words and generate a word cloud. Stopwords should be removed, and all words should be converted to lowercase. You may also choose to *stem* or *lemmative* your text, and may choose to restrict the word cloud to, e.g., the top 50 words if it is too cluttered.

   b. Repeat the above using *n-grams*. An *n-gram* corresponds to a set of *n* consecutive words. A *bigram* is an *n-gram* where *n = 2* and a *trigram* is an *n-gram* where *n = 3*. For example, the sentence "how are you?" has bigrams of "how are" and "are you". Using *textblob*, *n-grams* can be accessed using *blob.ngrams(n)*. Generate a table and/or wordcloud showing the most common bigrams and/or trigrams.

   c. Sentiment analysis (select at least one)

      i. Output the five most positive sentences (or lines), and the five most negative sentences (or tweets).

      ii. Compare the sentiment of two different text collections (e.g., two different Wikipedia pages, two different chapters, etc). Recommended: compare sentiments visually using pandas boxplot (https://wellsr.com/python/python-create-pandas-boxplots-with-dataframes/)

   d. Use named entity recognition to output the most commonly mentioned entities for various types, such as the most common people (PERSON), organizations (ORG) or geopolitical entities (GPE). Hint: use a *Counter* object to count the frequency of each element in a list (for an example see https://www.hackerrank.com/challenges/collections-counter/problem)

   e. Other analyses may be acceptable. I encourage you to be creative and contact me with ideas. I am also happy to help with data preparation. For example for a movie script, you may wish to compare the dialogue of two characters. In that case I can provide code for you to separate the dialogue for two characters.