**Sentiment Analysis**

**Data Mining and Warehousing**

**Rishabh Chawla**

**13BCE0379**

**Task**

Scrape, clean the scraped data of the speeches of a certain president and analyze all the speeches for their positivity and negativity based on more than context and also display the frequency of the words used.

**Dataset**

(<http://millercenter.org/president/speeches#washington>) was scraped for the speeches of George W Bush and then analyzed. [Present in the data folder]

**Background**

Sentiment analysis refers to the use of processing, text and [computational linguistics](https://en.wikipedia.org/wiki/Computational_linguistics) to identify and extract subjective information in source materials. Sentiment analysis is widely applied to reviews and social media for a variety of applications, ranging from [marketing](https://en.wikipedia.org/wiki/Marketing) to [customer service](https://en.wikipedia.org/wiki/Customer_relationship_management). A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature level — whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral.

This task is commonly defined as classifying a given text into one of two classes: objective or subjective This problem can sometimes be more difficult than polarity classification. The subjectivity of words and phrases may depend on their context and an objective document may contain subjective sentences. Moreover, as mentioned by Su, results are largely dependent on the definition of subjectivity used when annotating texts. However, Pang showed that removing objective sentences from a document before classifying its polarity helped improve performance.

**Algorithm**

**Scrape.py**: The speeches were scraped by using the python scraping tool(beautiful soup) and according to the class name they had. The scraped files are stored in the Data folder. It works in the manner that it gets the html code from an url and then goes to other urls from that url if needed else after getting the html code it traverses to the part which has the tags or the class names or the id names the user requires and then the data can be stored in variables or in files according to the requirement of the user.

**Clean.py**: After scraping if the data needed cleaning, the clean.py could be used to clean the file. It cleans special characters which come along with scraped data. It as well changes the codes of the special characters to the special characters as used in HTML. It adds new lines and makes the data more readable which removing as much noise as possible based on the mix the words are written as, or chunks of data written together.

**Freqcsv.py**: This script is used to count the occurances of words in a data file and arrange them in the ascending order. The output is stored in Freq.csv

**Data/Textblob.py**: This script uses textblob analysis to analyse all the files present in the data folder. It does sentiment analysis based on the phrases used and parts of the speech used.

**Textblobp3.py**: Does Sentiment Analysis based on the textblob library.

**Data1.py**: Does Sentiment Analysis based on the ntlk library with the data set provided. The data provided is in the positive.txt and negative.txt

**Platform**

**Python**

Python is a widely used [general-purpose](https://en.wikipedia.org/wiki/General-purpose_programming_language), [high-level programming language](https://en.wikipedia.org/wiki/High-level_programming_language) Its design philosophy emphasizes code [readability](https://en.wikipedia.org/wiki/Readability), and its syntax allows programmers to express concepts in fewer [lines of code](https://en.wikipedia.org/wiki/Lines_of_code). Python supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming) and [functional programming](https://en.wikipedia.org/wiki/Functional_programming) or  [procedural](https://en.wikipedia.org/wiki/Procedural_programming) styles. It features a [dynamic type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](https://en.wikipedia.org/wiki/Memory_management) and has a large and comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library).

**NTLK**

NLTK is a leading platform for building Python programs to work with human language data. Natural language processing (NLP) is a field of [computer science](https://en.wikipedia.org/wiki/Computer_science), [artificial intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence), and [computational linguistics](https://en.wikipedia.org/wiki/Computational_linguistics) concerned with the interactions between [computers](https://en.wikipedia.org/wiki/Computer) and [human (natural) languages](https://en.wikipedia.org/wiki/Natural_language). As such, NLP is related to the area of [human–computer interaction](https://en.wikipedia.org/wiki/Human%E2%80%93computer_interaction).

**TEXTBLOB**

TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

**BEAUTIFUL SOUP**

[Beautiful Soup](http://www.crummy.com/software/BeautifulSoup/) is a Python library for pulling data out of HTML and XML files.

**Experiment Results and Findings**

The scraped data is present in the “Data” folder with its analysis. The programs are attached.

Results for the file used for example purposes:

Most frequently used word: University (196)

Positivity from textblob: 0.2035(analysis.txt)

Positivity from dataset: 0.2425(tweet\_sentiment.csv)

Cleaned files are clean2.txt and clean3.txt

**Findings**: The US presidents try to keep their negative speeches overall neutral.