 Due by midnight today

CMPSC 102
Discrete Structures
Fall 2019

Practical 13: Analysis of Text

Refer to your notes, slides and sample Python code from this week and other weeks. In particular, follow the python code that we created in class or check on line for interesting pieces of code to help you in your programming.

Figure 1: There are many ways to discover the meaning in text without reading manually.

GitHub Starter Link

https://classroom.github.com/a/kQ-36CN9

To use this link, please follow the steps below.

- Click on the link and accept the assignment.
- Once the importing task has completed, click on the created assignment link which will take you to your newly created GitHub repository for this lab.
- Clone this repository (bearing your name) and work on the practical locally.
- As you are working on your practical, you are to commit and push regularly. You can use the following commands to add a single file, you must be in the directory where the file is located (or add the path to the file in the command):
  - `git commit <nameOfFile> -m ‘‘Your notes about commit here’’`
  - `git push`

  Alternatively, you can use the following commands to add multiple files from your repository:
  - `git add -A`
  - `git commit -m ‘‘Your notes about commit here’’`
  - `git push`

Handed out: 2nd December 2019

Summary

In the Internet age, there is seemingly a never-ending supply of information coming from all over the online landscape. In the past, companies and other organizations were able to employ humans to read through their received information from customers and markets to gain feedback. However, due to the massive amounts of information they receive, many organizations today are no-longer able manually read all communications and so machines are used to collect, sort and digest the information from customers, discussed in Figure 1.

Figure 2: A document may have an optimistic, neutral or pessimistic nature when analyzed by sentiment of words. There are other types of analysis that we have covered during Discrete Structures CS102.

Sentiment analysis is one of the leading forms of analysis that organizations use to determine the *gist* of their customer mails – a mail's pessimistic or optimistic score helps to prioritize it. Sentiment analysis has become a very powerful tool across many types of businesses, you will spend some time learning how this form of analysis (generally) works by playing with a simple program to study some samples of text. Here, we note that sentiment analysis has become able to successfully detect the levels of optimism, neutrality or pessimistic (discussed in Figure 2) in text thanks to also applying the text to other types of statistical tests.

In this practical, you will be studying some code that completes a basic sentiment analysis of text. Try typing a sentence into the program (as one of its inputs) to learn more about its functionality. You will enrich this analysis by adding your own mathematical and statistical programming to the code to complete other forms of analyses.

The Steps to Complete

- Take a moment to play with the code in your repository and to learn how it works with the included sample text files.
- Then add two new statistical functions to the code to collect word frequencies and then to plot these frequencies.
- Open challenge: add another type of statistical test to the code to increase your understanding of the text without having to read it. This test can be *any test* you like as long as it produces some result to learn about the text in some way. For example, can you analyze the frequencies of the sentiment words?

Code Usage

There are two ways to enter textual data into the `sentiminer.py`program: by a typed sentence and by a file. Running the program without an parameters will give the below online help.

```
+ To enter a sentence manually: python3 sentiminer_practical13.py s
+ To enter a file: python3 sentiminer_practical13.py f filename.txt
```

Deliverables

1. File `src/sentiminer_practical13.py`: Your completed code and an output markdown file (please create the file, `src/output.md`) to showcase the output of your added functions: `wordFrequencies()` and `frequencyPlotter()`. This output files will also showcase your added challenge function that completes some other statistical analysis or treatment of the text. Note: please include comments and `print` statements in your code to add documentation and to help the user understand your program's tests.

.... --- .-- / .... .- .--. .--. -.-- / .. ... / -.-- --- ..- .-. / - . -..- - ..--..