Bad File

PROJECT 4

PREPARED FOR

Dr. Sabine Bergler

PREPARED BY

Karandeep

40104845

TABLE OF CONTENTS

Introduction:

About NLTK - NLTK aka the Natural Language Toolkit, is a suite of open source Python modules, data sets, and tutorials supporting research and development in Natural Language Processing. It contains text processing libraries for tokenization, parsing, classification, stemming, tagging and semantic reasoning. It also includes graphical demonstrations and sample data sets as well as accompanied by a cook book and a book which explains the principles behind the underlying language processing tasks that NLTK supports.

Purpose:

Hidden information often lies deep within the boundaries of what we can perceive with our eyes and our ears. Some look to data for that purpose, and most of the time, data can tell us more than we thought was imaginable. But sometimes data might not be clear cut enough to perform any sort of analytics. Language, tone, and sentence structure can explain a lot about how people are feeling, and can even be used to predict how people might feel about similar topics using a combination of the Natural Language Toolkit, a Python library used for analyzing text, and machine learning.

For this project I was to develop a Feature grammar to parse given sentences and judge the sentiment behind the sentence. The console has options to run **Sentiment Analyzer** either on a good inbuilt data or bad inbuilt data sentences to parse. Analysis of data has been done in a way that is easily comprehensible. Overall goal of the project was to create a parser and a FCFG which filters data by using NLTK access commands and creates sentence trees and visualize them in a more informative way. NLTK is not perfect and lacks in some areas which is being explored below.

Bad Validation Data (Feature Context Free Grammar)/ Sentiment Parser:

Examples:

- Bad Sentence 1
  OUTPUT:

```
Sentence ->  it is not too much bad to recommend

(S[-INV]
  (NP[-WH] it)
  (VP[]
    (V[+AUX]
      (V[+AUX] (V[+AUX] is) (ADVB[+NEG] not))
      (ADVB[-NEG] too))
    (NP[]
      (NP[]
        (NP[] (ADVB[-NEG] much))
        (NP[SNT='NEG'] (ADJ[SNT='NEG'] bad)))
      (NP[] (PRP[] to) (NP[SNT='POS'] (ADJ[SNT='POS'] recommend))))))

          ****************************************
          #     Project 3 Output: The sentence is       Negative
          #     SSAP Baseline Output: The sentence is   Negative
          #     Project 4 Output: The sentence is       Negative
          ****************************************
```

- Bad Sentence 2
  OUTPUT:

```
Sentence ->  Apple Music is not really an amazing service

(S[-INV]
  (NP[] (NP[-WH] Apple) (NP[-WH] Music))
  (VP[]
    (V[+AUX]
      (V[+AUX] (V[+AUX] is) (ADVB[+NEG] not))
      (ADVB[-NEG] really))
    (NP[]
      (NP[]
        (NP[-WH] (DT[] an))
        (NP[SNT='POS'] (ADJ[SNT='POS'] amazing)))
      (NP[-WH] service))))

          ****************************************
          #     Project 3 Output: The sentence is       Neutral
          #     SSAP Baseline Output: The sentence is   Positive
          #     Project 4 Output: The sentence is       Positive
          ****************************************
```

- Bad Sentence 3
  OUTPUT:

```
Sentence ->  it is not completely his fault that it is not right

(S[-INV]
  (NP[-WH] it)
  (S[-INV]
    (V[+AUX] (V[+AUX] is) (ADVB[+NEG] not))
    (NP[]
      (NP[]
        (NP[] (ADVB[+NEG] completely))
        (NP[]
          (NP[] (NP[-WH] his) (NP[SNT='NEG'] (ADJ[SNT='NEG'] fault)))
          (Comp[] that)))
      (NP[-WH] it))
    (VP[]
      (V[+AUX] is)
      (NP[] (ADVB[+NEG] not))
      (ADJ[SNT='POS'] right))))

          ************************************
          #     Project 3 Output: The sentence is       Negative
          #     SSAP Baseline Output: The sentence is   Neutral
          #     Project 4 Output: The sentence is       Negative
          ************************************
```