Sentiment Analysis on the NLTK Corpus of Movie Reviews using Python

LING 571 : Corpus Linguistics - Programming Project

Bhagyashri Patil

San Diego State University

Address: 5485 55th St, San Diego, CA 92115

Contact: bpatil0085@sdsu.edu

Cell: (619) 844-3051

December 25, 2021

Introduction

In the last decade, sentiment analysis has become more popular, as day-by-day there is increasing number of users using the social media platforms, surveys, feedbacks, live platforms, etc. to express their opinions. Sentiment Analysis can be defined as the combined usage of Text analysis, Natural language processing *(NLP)*, and Computational Linguistics to consistently recognize, draw out, measure, and learn emotional states and personal information. The applications of Sentiment Analysis are online reviews, surveys, social media comments, marketing campaign details, customer service data, health science preferences, etc.

In this programming project, we will be considering the Movie Reviews corpus provided by the Natural Language ToolKit *(NLTK)* website. NLTK is an open-source platform which provides a great variety of libraries and functions for statistical analysis in the natural language processing. NLTK helps in supporting and solving various domains such as artificial intelligence, machine learning, information recovery, psychological science, etc.

Sentiment analysis does the primary task of determining whether the tone of the comment, feedback, review, or response is either positive or negative or if its neutral (mixed). It can find this at various granularities such as words, sentence, chapter, or document. Further, we will focus on the sentiment analysis functions and method for implementing it using python.

Data and Methods

- Python Code File:

https://colab.research.google.com/drive/13poIaJ6IuV0hUMJIfQDuVt1dAtvVkpjw?usp=sharing

- Importing NLTK and Movie Reviews Corpus:

```
import nltk
nltk.download('book', quiet=True)
from nltk.corpus import movie_reviews
```

- Looking for words in the Movie Reviews corpus:

```
movie_reviews.words()
>> ['plot', ':', 'two', 'teen', 'couples', 'go', 'to', ...]
```

- Check the size of the Movie Reviews corpus:

```
len(movie_reviews.words())
>> 1583820
```

- Looking for the Categories of Movie Reviews corpus which gives us the type of reviews:

```
movie_reviews.categories()
>> ['neg', 'pos']
```

- Calculating the Frequency Distribution of the words in the Movie Reviews corpus:

```
movie_reviews_freq = nltk.FreqDist(movie_reviews.words())
movie_reviews_freq.most_common(20)

>> [(',', 77717),
 ('the', 76529),
 ('.', 65876),
 ('a', 38106),
 ('and', 35576),
 ('of', 34123),
 ('to', 31937),
 ("'", 30585),
 ('is', 25195),
 ('in', 21822),
 ('s', 18513),
 ('"', 17612),
 ('it', 16107),
 ('that', 15924),
 ('-', 15595),
 (')', 11781),
 ('(', 11664),
 ('as', 11378),
 ('with', 10792),
 ('for', 9961)]
```

- Looking for count of Distinct words in the Movie Reviews corpus:

```
len(movie_reviews_freq)
>> 39768
```

- Checking for the File IDs of the Movie Reviews corpus:

```
movie_reviews.fileids()

>> ['neg/cv000_29416.txt',
 'neg/cv001_19502.txt',
```

'neg/cv002_17424.txt',
'neg/cv003_12683.txt',
'neg/cv004_12641.txt', …]

Here, we can see the Fileids have the starting as 'neg' and 'pos'. So, this indicates that every file id is denoted by either positive or negative tag.

- Looking for the Reviews with Positive tag:

  movie_reviews.fileids('pos')

  >> ['pos/cv000_29590.txt',
  'pos/cv001_18431.txt',
  'pos/cv002_15918.txt',
  'pos/cv003_11664.txt',
  'pos/cv004_11636.txt', …]

- Let's check for the words containing in this Positive tagged fileids:

  movie_reviews.words('pos/cv000_29590.txt')
  >> ['films', 'adapted', 'from', 'comic', 'books', 'have', ...]

- Looking for the Reviews with Negative tag:

  movie_reviews.fileids('neg')

  >> ['neg/cv000_29416.txt',
  'neg/cv001_19502.txt',
  'neg/cv002_17424.txt',
  'neg/cv003_12683.txt',
  'neg/cv004_12641.txt', …]

- Let's check for the words containing in this Negative tagged fileids:

  movie_reviews.words('neg/cv000_29416.txt')
  >> ['plot', ':', 'two', 'teen', 'couples', 'go', 'to', ...]

- Creating master list of all the words contained in the frequency distribution of the Movie Reviews corpus:

  movie_reviews_list = list(movie_reviews_freq)[:4000]
  movie_reviews_list

  >> ['plot',
  ':',

'two',
'teen',
'couples',
'go',
'to',
'a',
'church',
'party', …]

- First step for analyzing the Sentiments of Movie Reviews:

```
movie_reviews = {}
```

- I have chosen a single movie, to pass for the sentiment analysis:

```
reviews = nltk.corpus.movie_reviews.words('neg/cv942_18509.txt')
```

If any word from 'movie_reviews_list' matches with the 'reviews' then we give output as TRUE, else FALSE.

```
for i in range(len(movie_reviews_list)):
  movie_reviews[movie_reviews_list[i]] = movie_reviews_list[i] in reviews
```

- Looking for all words to whom we have alloted TRUE:

```
[i for i in movie_reviews_list if movie_reviews[i] == True]
```

```
>> ['plot',
':',
'to',
'a',
',',
'and', …]
```

Here, we can observe the works such as 'insults', 'destroys', 'attacks' etc. which conclude that the sentiment of the file is Negative.

- Creating a Document which comprises of the words and categories of the reviews:

```
movie_reviews_document = [
                    (nltk.corpus.movie_reviews.words(reviews_file_id), cat)
                    for reviews_file_id in nltk.corpus.movie_reviews.fileids()
                    for cat in nltk.corpus.movie_reviews.categories(reviews_file_id)
                    ]
```

- Displaying Document contents:

```
movie_reviews_document
>> [(['plot', ':', 'two', 'teen', 'couples', 'go', 'to', ...], 'neg'),
 (['the', 'happy', 'bastard', '"', 's', 'quick', 'movie', ...], 'neg'),
 (['it', 'is', 'movies', 'like', 'these', 'that', 'make', ...], 'neg'),
 (['"', 'quest', 'for', 'camelot', '"', 'is', 'warner', ...], 'neg'),
 (['synopsis', ':', 'a', 'mentally', 'unstable', 'man', ...], 'neg'), ...]
```

Results

- Defining a function which searches the required TRUE/FALSE sentiments:

```
def search_sentiments(all_words):
  sentiment_function = {}
  for i in movie_reviews_list:
    sentiment_function[i] = i in all_words

  return sentiment_function
```

- Displaying results for the function search_sentiments for the very first file in the document created:

```
search_sentiments(movie_reviews_document[1][0])
>> {'plot': False,
 ':': False,
 'two': False,
 'teen': False,
 'couples': False,
 'go': False,
 'to': True,
 'a': True,
 'church': False,
 'party': False,
 ',': True, …]
```

Abstract

My main objective for selecting this topic was very clear. If we have got any dataset comprising of the reviews or feedbacks or comments or responses, will we be able to identify the tone of the sentiments by using the nltk corpus dataset and python functions? I finalized on this topic because it is quite intriguing to understand and learn the methods and procedures to analyze

the each and every sentiment and to come with a conclusion regarding its tone and usage. The functions I have used in this paper, are related to the ones I have studied in one of my courses of Corpus Linguistics. My aim with this programming topic was to understand the functioning of the Sentiment analysis functions and in what possible way will it affect the analysis of the various textual content in the nearby future. I have referred the NLTK corpus of the Movie Reviews consisting of reviews of various movies, to establish my findings and perform some analysis on the textual data present in this dataset and to present my results and conclusions on my programming. When we speak of public opinions, we don't have the guessing freedom of what exactly a person will express or mention. Now-a-days there are so many portals or interfaces, where people can openly express their opinions regarding any movie released, or any particular controversial scene shot in that movie. Also, while submitting the feedback or reviews, hardly the people will check if they have spelled everything correctly or added punctuations correctly. So we also need to consider this factors while considering the movie reviews and design our sentiment function accordingly.

Conclusion

There are various platforms who provide the movie reviews such as IMDb, Rotten Tomatoes, Film.com etc. which provide with the detailed public reviews regarding the movies released recently and in old times. For any specific movie, there are huge number of reviews and responses generated from the viewers.

When any person decides to watch any movie, he first goes to these portals and goes through the reviews to confirm if he wants to go ahead with that movie or not. However, it will not be always possible that he will get clear idea after reading the reviews, as the reviews can be most of the times neutral i.e., 50% positive and 50% negative. This can often mislead him in taking his decision.

For solving this, we have the sentiment analysis technique which can sum up and give brief conclusion to any person, for helping him decide on to which movie he should finalize to watch. The programming from this project can be useful to understand the base math theory and the purpose behind developing sentiments analysis. From the results and the conclusions of our

sentiment analysis, we will be able to clearly identify the tone of the reviews and come towards a generic conclusion by using the python libraries and functions.

References

[1] Natural Language Processing with Python,
https://www.nltk.org/book/
[2] Sentiment Analysis – *Wikipedia*,
https://en.wikipedia.org/wiki/Sentiment_analysis