Ricardo Godoy

CS2302

Lab Report

Introduction

The purpose of this program is to access a reddit post and recursively iterate through all the comments, while reading them. As the program reads each comment, it will call for methods get_text_negative_proba(), get_text_neutral_proba(), and get_text_positive_proba() methods to test its negativity, neutrality, and positivity. These three methods will return a value between 0 to 1, 1 being the maximum negative, neutral, or positive value respectively. Once these values are obtained, the program will compare the values returned and decide in what list should each comment be directed to. The program will return three lists:
negative_comments_list, containing negative comments

neutral_comments_list, containing neutral comments

positive_comments_list, containing positive comments.

Proposed solution design

I intended to approach the comments and their replies as if it was a tree. Started from the top comment, seen as the root, the program traverses through the comment's replies, which are the leaves, and the replies' replies, and so on. Approaching the comments as a tree perfectly worked to iterate recursively through all the replies (leaves) and for each leave, a call to the methods to analyze its sentiment was made. I designed the program to work with a list of comments of different lengths. The only base case that it was used was if the length of the list of comments was 0, in which case the program will only return 0. Any list of value greater or equal than 1 will run perfectly.

Experimental Results

I decided to experiment with different posts I found on reddit, with different types and length of comments. This was due in order to test the program with different types of inputs and observe the output. The process for each post was exactly the same. The program would read the comments and iterate through them, following the tree approach. After reading each comment, the program would call the three methods, each one returning a value between 0 and 1, depending of the sentiment of the comment. Then, I compared all three values obtained to determine the sentiment of the comment and therefore, append it into its corresponding list. First, I used a reddit post with 200 comments and replies total. The program demonstrated that it was able to iterate through all the comments and leaves with no issues. Along with that, I was able to observe that after printing the length of all three lists to make sure the lists had object in them, I noticed that the total of all three lists added up to 199. Probably it was due to one or more if statements. This was the obtained output:

Negative: 2

Positive: 11

Neutral: 186

Another post I decided to use: (https://www.reddit.com/r/hmmm/comments/9gy5mm/hmmm/) with a total of 147 comments and replies total (up to date) gave me a total of 144 comments on all lists. The output I obtained from this test was:

Negative: 6

Positive: 5

Neutral: 133

The running time of my program is O(n). It depends on the size of the tree (number of comments and replies).

Conclusion

From this program, I strengthened my skills on how to recursively traverse not only through the comments of a reddit post, but also through a tree. I learned how to approach different problems and along with that, I also learned the Python syntax,

which before this class, it was unknown to me. Perhaps I need to strengthen and increase my knowledge on if statements and how to calculate the running time of different algorithms.

Source code

```
#Ricardo Godoy

#This program reads reddit comments and replies from any post and classifies them as neutral,

#positive, or negative comments, depending of the words used in each comment or reply.

#Last date modified: 9/17/18

import nltk

from nltk.sentiment.vader import SentimentIntensityAnalyzer

import praw


reddit = praw.Reddit(client_id='UdOgsdFvN5vfig',

                     client_secret='fKBTGDzr7PsDkdG5GENs--8efVU',

                     user_agent='Recursion'

                     )



nltk.download('vader_lexicon')

sid = SentimentIntensityAnalyzer()



def get_text_negative_proba(text):
```

```
    return sid.polarity_scores(text)['neg']


def get_text_neutral_proba(text):
    return sid.polarity_scores(text)['neu']


def get_text_positive_proba(text):
    return sid.polarity_scores(text)['pos']


def get_submission_comments(url):
    submission = reddit.submission(url=url)
    submission.comments.replace_more()

    return submission.comments

#This method recursively navigates through the list of comments generated above, and appends
#each comment into its corresponding list.
def process_comments(comment, neglist, neulist, poslist):

    if comment is None:
        return 0
```

```
    for i in range(len(comment)):
            neg = get_text_negative_proba(comment[i].body)
            neu = get_text_neutral_proba(comment[i].body)
            pos = get_text_positive_proba(comment[i].body)
            if (neg > pos and neg > neu ):
               neglist.append (comment[i])
            elif (pos > neg and pos > neu):
               poslist.append (comment[i])
            elif (neu > pos and neu > neg):
               neulist.append (comment[i])
            process_comments(comment[i].replies, neglist, neulist, poslist)


    return neglist, neulist, poslist


def main():
    comments =
get_submission_comments('https://www.reddit.com/r/learnprogramming/commen
ts/5w50g5/eli5_what_is_recursion/')
    negative_comments_list = []
    positive_comments_list = []
    neutral_comments_list = []
    negative, neutral, positive = process_comments(comments,
negative_comments_list, neutral_comments_list, positive_comments_list)
    print("negative: " + str(len(negative_comments_list))
    print("positive: " + str(len(positive_comments_list)))
    print("neutral: " +str(len(neutral_comments_list)))
```

main()