Lab 1 Report

The main problem that I am trying to solve is a reddit content that will determine a chosen post and determine whether the comments, sub-comments, or replies are negative, positive, or neutral comments. I solve the problem using recursion. Recursion will allow me to call a method again until it is completely executed. In this case through recursion the method will analyze where the comments belong. (Different categories.) One of the problems I presented was to determine on which order I was going to check for the values and how will I determine if a comment was either positive, negative, or neutral. In order to solve it I had to break down the problem into pieces. For example, I created a list of arrays where these comments were going to be stores once they were executed by the recursive method. Another step that I took was to set a value of .50 were the comment will be identified as neutral. Lastly, but not least I provided a method that will allow to see if there were more comments or replies additionally to the original comments which would be categorized as well. I definitely learned from this project that the use of recursion is a very effective and powerful method or idea that follows a specific pattern until it reaches a base case ending the method.

# CS2302 Data Structures  
# Use of Recursion in order to determine and analyze whether a section of comments  
# Fit into positive, negative, or normal category.  
# Programmed by Luis Garcia.  
# Last modified September 18, 2018.  
# Instructor Diego Aguirre.  
# Lab1  
  
import nltk  
from nltk.sentiment.vader import SentimentIntensityAnalyzer  
import praw  
  
reddit = praw.Reddit(client\_id='vo3B1Fd51rupXg', # To obtain the client id I went into reddit and create an application.  
 client\_secret='vrJEX\_48mRmiouINsUfJUxAtRH4', # Client Secret was also created through reddit.  
 user\_agent='lgarcia27' # This user agent is the one I created on reddit.  
 )  
  
nltk.download('vader\_lexicon')  
sid = SentimentIntensityAnalyzer()  
  
# I created a group of lists which will be kept in an array.  
normalList = [] # The Normal List will store the normal comments of reddit.  
negativeList = [] # The Negative List will store the negative comments of reddit.  
positiveList = [] # The Positive List will store the positive comments of reddit.  
  
# My Code  
# This method helps to test and classified whether a comment is negative or positive  
# with the use of a normal comment as a value.  
def classificationComments(comment, x):  
 neg = get\_text\_negative\_proba(comment.body)  
 pos = get\_text\_positive\_proba(comment.body)  
 if neg-x > pos-x:  
 positiveList.append(comment.body) #Append will add items to the end of the list.  
 return negativeList.append(comment.body)  
  
  
# This method will support if there is a reply to a comment or comments and determine  
# if the comment will go as normal.  
def process\_comments(x):  
 if len(x) == 0 or x is None:  
 return [],[],[]  
  
 positive, neutral, negative = [],[],[]  
  
 # TESTING  
 #print(x[0].body)  
 #print()  
  
 w = get\_text\_negative\_proba(x[0].body)  
 y = get\_text\_neutral\_proba(x[0].body)  
 z = get\_text\_positive\_proba(x[0].body)  
  
 prob = max(w,y,z)  
  
 # IF the max is negative  
 if prob == w:  
 negative.append(x[0].body)  
 # If the max is neutral  
 elif prob == y:  
 neutral.append(x[0].body)  
 # If the max is positive  
 else:  
 positive.append(x[0].body)  
  
 # Go to replies of current node  
 # Temporarily store lists  
 posT, neuT, negT = process\_comments(x[0].replies)  
 # Add reply elements to lists  
 positive.extend(posT)  
 neutral.extend(neuT)  
 negative.extend(negT)  
  
 # Process same-level comments  
 posT, neuT, negT = process\_comments(x[1:])  
 # add reply element to lists  
 positive.extend(posT)  
 neutral.extend(neuT)  
 negative.extend(negT)  
  
 return positive, neutral, negative  
  
# Code given  
def get\_text\_negative\_proba(text):  
 return sid.polarity\_scores(text)['neg']  
  
  
def get\_text\_neutral\_proba(text):  
 return sid.polarity\_scores(text)['neu'] # I used normal instead of neutral  
  
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  
  
  
def main():  
 comments = get\_submission\_comments('https://www.reddit.com/r/politics/comments/9glw1w/avenatti\_on\_possible\_2020\_presidential\_run\_im/')  
 # comments2 = get\_submission\_comments()  
 posList, neuList, negList = process\_comments(comments)  
  
 #Printing lists  
 print("Positive list: ", posList, "\n")  
 print("Neutral list: ", neuList, "\n")  
 print("Negative list: ", negList, "\n")  
  
  
 '''  
 #line 73 provides a link which will acess the library of the comments and will help with testing purposes.  
 #The reason I choose this URL is because I believe politics is a very controversial toppic where you can always find positive, neutral, and negative comments.  
 # As far as testing purposes it is relevant.  
 print(comments[0].body)  
 print(comments[0].subComments[0].body)  
 #print statements will help in maintaining the order of the comments rather than having all over the place.  
 #print(positiveList)  
 #// print('Positive Comments') # This print line will help into the organization of the code and will print "Positive   
 #Comments in order to follow easier were these specifics comments are.  
 #print(negativeList)  
 #//print('Negative Comments') # This print line will help into the organization of the code and will print "Negative   
 #Comments in order to follow easier were these specifics comments are.  
 #print(normalList)  
 #//print('Normal Comments') # This print line will help into the organization of the code and will print "Normal   
 #Comments in order to follow easier were these specifics comments are.  
  
 neg = get\_text\_negative\_proba(comments[0].subComments[0].body)  
  
 print(neg)'''  
  
main()

“I certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in class.”