**Mô tả**

1. Đánh giá quan cảm của bình luận

Việc khai thác dữ liệu ý kiến (Opinion Mining) là kỹ thuật được sử dụng để xác định quan cảm, cảm xúc trong một dữ liệu văn bản. Bình luận trên Youtube là nơi người xem thể hiện đánh giá của mình với nội dung video đó.

Khai báo package Python

Trong đó:

- nltk.sentiment.vader là package công cụ phân tích quan cảm

import numpy as np  
import pandas as pd  
from nltk.sentiment.vader import SentimentIntensityAnalyzer  
import matplotlib.pyplot as plt

Tạo object SentimentIntensityAnalyzer

sia = SentimentIntensityAnalyzer()

Đọc file dữ liệu

df = pd.read\_csv('FLAT.csv', encoding\_errors = 'ignore')

Thêm dữ liệu điểm đánh giá

Tạo các cột dữ liệu liên quan đến điểm tổng thể, điểm tích cực, điểm tiêu cực

df['scores'] = df['comment'].apply(lambda body: sia.polarity\_scores(str(body)))  
df.head()  
df['compound'] = df['scores'].apply(lambda score\_dict:score\_dict['compound'])  
df.head()  
df['pos'] = df['scores'].apply(lambda pos\_dict:pos\_dict['pos'])  
df.head()  
df['neg'] = df['scores'].apply(lambda neg\_dict:neg\_dict['neg'])  
df.head()

Phân loại tính chất bình luận

Tạo cột dữ liệu tên “type”, dùng để phân loại tính chất của binh luận:

- POS: Tích cực

- NEG: Tiêu cực

- NEUTRAL: Trung tính

df['type']=''  
df.loc[df.compound>0,'type']='POS'  
df.loc[df.compound==0,'type']='NEUTRAL'  
df.loc[df.compound<0,'type']='NEG'  
df.head()

Đếm số lượng bình luận theo tính chất

len=df.shape  
(rows,cols)=len  
pos=0  
neg=0  
neutral=0  
for i in range(0,rows):  
 if df.loc[i][5]=="POS":  
 pos=pos+1  
 if df.loc[i][5]=="NEG":  
 neg=neg+1  
 if df.loc[i][5]=="NEUTRAL":  
 neutral=neutral+1

print("Positive :"+str(pos) + " Negative :" + str(neg) + " Neutral :"+ str(neutral))

Tạo biểu đồ

plt.hist(df['compound'])  
plt.title("Histogram")  
plt.show()  
  
print("Average Positive: ", df['pos'].mean(), "\n")  
print("Average Negative: ", df['neg'].mean(), "\n")  
print("Average Compound: ", df['compound'].mean(), "\n")

2. Xử lý dữ liệu nâng cao

Xử lý dữ liệu bao gồm những công việc như loại bỏ từ dừng, loại bỏ những ký tự đặc biệt hay đường link để có thể phân tích chính xác hơn. Sau đó văn bản được chuyển về dạng viết thường, Tokenization, Stemming và Lemmatization để phân tích dữ liệu rõ hơn

Khai báo package Python

import re  
import numpy as np  
import pandas as pd  
import nltk

Đọc file và tạo cột dữ liệu detail để xử lý

df = pd.read\_csv('FLAT.csv', encoding\_errors = 'ignore')  
df['detail'] = df['comment']  
df.head()  
  
df['detail'] = df['detail'].str.lower()  
df.tail()

Tạo set chứa từ dừng trong tiếng Anh

clean\_word = ['a', 'about', 'above', 'after', 'again', 'ain', 'all', 'am', 'an',  
 'and','any','are', 'as', 'at', 'be', 'because', 'been', 'before',  
 'being', 'below', 'between','both', 'by', 'can', 'd', 'did', 'do',  
 'does', 'doing', 'down', 'during', 'each','few', 'for', 'from',  
 'further', 'had', 'has', 'have', 'having', 'he', 'her', 'here',  
 'hers', 'herself', 'him', 'himself', 'his', 'how', 'i', 'if', 'in',  
 'into','is', 'it', 'its', 'itself', 'just', 'll', 'm', 'ma',  
 'me', 'more', 'most','my', 'myself', 'now', 'o', 'of', 'on', 'once',  
 'only', 'or', 'other', 'our', 'ours','ourselves', 'out', 'own', 're','s', 'same', 'she', "shes", 'should', "shouldve",'so', 'some', 'such',  
 't', 'than', 'that', "thatll", 'the', 'their', 'theirs', 'them',  
 'themselves', 'then', 'there', 'these', 'they', 'this', 'those',  
 'through', 'to', 'too','under', 'until', 'up', 've', 'very', 'was',  
 'we', 'were', 'what', 'when', 'where','which','while', 'who', 'whom',  
 'why', 'will', 'with', 'won', 'y', 'you', "youd","youll", "youre",  
 "youve", 'your', 'yours', 'yourself', 'yourselves']  
cleaner = set(clean\_word)

Loại bỏ từ dừng

def cleaning(text):  
 return " ".join([word for word in str(text).split() if word not in cleaner])  
  
df['detail'] = df['detail'].apply(lambda text: cleaning(text))  
df.head()

Loại bỏ dấu câu

import string  
english\_punctuations = string.punctuation  
punctuations\_list = english\_punctuations  
def cleaning\_punctuations(text):  
 translator = str.maketrans('', '', punctuations\_list)  
 return text.translate(translator)  
df['detail']= df['detail'].apply(lambda x: cleaning\_punctuations(x))  
df.tail()

Loai bỏ từ lặp

def cleaning\_repeating\_char(text):  
 return re.sub(r'(.)1+', r'1', text)  
df['detail'] = df['detail'].apply(lambda x: cleaning\_repeating\_char(x))  
df.tail()

Loại bỏ đường link

def cleaning\_URLs(data):  
 return re.sub('((www.[^s]+)|(https?://[^s]+))',' ',data)  
df['detail'] = df['detail'].apply(lambda x: cleaning\_URLs(x))  
df.tail()

Loại bỏ số

def cleaning\_numbers(data):  
 return re.sub('[0-9]+', '', data)  
df['detail'] = df['detail'].apply(lambda x: cleaning\_numbers(x))  
df.tail()

Tokenize

from nltk.tokenize import word\_tokenize  
df['detail'] = df['detail'].apply(lambda x: word\_tokenize(x))  
df.head()

Stemming

Đưa tử về dạng gốc trước khi được chia

st = nltk.PorterStemmer()  
def stemming\_on\_text(data):  
 text = [st.stem(word) for word in data]  
 return data  
df['detail'] = df['detail'].apply(lambda x: stemming\_on\_text(x))  
df.head()

Lemmatization

Đưa từ về gốc từ của nó

lm = nltk.WordNetLemmatizer()  
def lemmatizer\_on\_text(data):  
 text = [lm.lemmatize(word) for word in data]  
 return data  
df['detail'] = df['detail'].apply(lambda x: lemmatizer\_on\_text(x))  
df.head()