Sentiment Analysis with Artificial Intelligence Using Python modules – TextBlob and Emoji

Praveena Pillala

Research Scholar, Department of Computer Science and Engineering, Centurion University of Technology and Management, Visakhapatnam, Andhra Pradesh, India

Abstract:

Sentiment analysis or opinion mining is the computational linguistics of attitudes, emotions, issues, events, topics, and their attributes. Sentiment analysis plays a key role in the field of text mining. The objective of the sentiment classification is to investigate the subjective information that categorizes the polarity of the text (positive, negative and neutral).The purpose of this paper is to analyze and assign a polarity to examine the text.

Keywords: *Natural Language Processing, Sentiment Analysis, TextBlob, Emoji, Artificial Intelligence*

1. Introduction:

In the recent ages research has emerged significantly in Sentiment Analysis. The art of study that remained as the cornerstone to bridge communication between man and machine is termed as Natural Language Processing (NLP).NLP is a subfield of AI in whatever place a data processor can investigate, acknowledge and extract definition from the human communication in the form of verbal and nonverbal languages in a sophisticated and beneficial way.

With the help of NLP, programmers/researchers can heap up and organize data to execute responsibilities such as automatic summarization, translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition and topic segmentation. NLP considers the hierarchical composition of language: several words make a idiom, several phrases make a sentence and ultimately sentences convey thoughts. By studying the language for its definition, NLP have many useful roles, such as correcting grammar, converting speech to text and automatically translating between languages.

NLP is used to analyze texts, allowing machines to understand how human does speak. This human-computer interaction enables real-world applications like automatic text summarization, sentiment analysis, topic extraction, named entity recognition, parts-of-speech, tagging, relationship extraction, stemming and more. NLP is commonly used for text mining, machine translation and automated question answering.

NLP is characterized as a solution to the tricky problems in the arena of computer science. Human language is rarely defined or plainly spoken. To understand human language does not only understand the words but the concepts and how they're linked together to each other in creating meaning. Despite language being one of the easiest things for the human mind to learn, the ambiguity of language is what makes natural language processing a difficult problem for computers to master.

Sentiment Analysis is a subfield of natural language processing that analyses people's opinion/public opinion towards different products entities on social media and then checks the polarity of the text. Sentiment Analysis is also known as "Subjective Analysis" as it focuses on the subjective part of the text, phrase or sentence to know the direction of the text (good/bad).Sentiment analysis is the process of examining the emotional value in a series of word or text, to gain an understanding of the emotions expressed. Sentiment analysis can be applied in various sectors like ecommerce, banking, social websites like Face book, Twitter and so

on. One of the applications of sentiment analysis is recommendation systems, for example – YouTube based on likes, dislikes and comments given by the user.

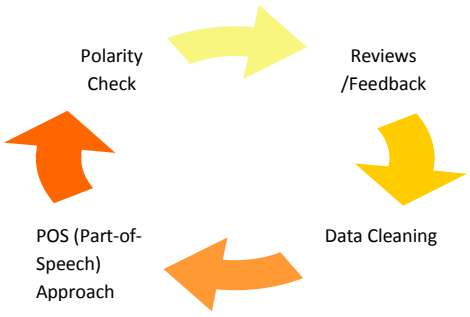

Figure 1: Framework: Sentiment Classification /Analysis

Sentiment Classification/Analysis and text mining mainly includes following subtasks as shown in figure 1. First, User feedback or reviews are collected based on different contents from social media. Second, preprocessing step is applied to clean the data (to remove unnecessary data).Thirdly, by using the part-of-speech (POS) approach required data is identified. Finally, Sentiment Classification is performed by using machine learning mechanisms, feature selection and applying machine learning algorithms in order to check the polarity of the text. Emoji and emoticon sentiments are applied in many studies to improve the precision of results and so on.

Most of the research exists on sentiment analysis for user opinion data, which mainly focuses the polarity of user reviews [1].

2. Levels of Sentiment Analysis:

Three different levels of sentiment analysis are mainly explored: Document Level, sentence level and aspect level. In document level reviews or feedback are figured out either as positive or negative. In sentence level every sentence is identified as positive or negative and in aspect level entities and aspects and their features is positive or negative.

2.1. Document Level:

In document level the entire task holds opinion of the group. The file/task verifies whether it conveys the positive or negative sentiment [2, 3].

2.2. Sentence Level:

Here the task is based on sentences and determines each one sentence is positive, negative or neutral. And it is not suitable for complex sentences/statements [4].

2.3. Entity and Aspect Level:

Entity and Aspect level analysis are also known as phrase level sentiment analysis. It verify the emotions which are present both at document level and the sentence level but do not discover what exactly user liked or disliked. Aspect level looks at the opinion itself, which identifies the feature values of the opinion and then summarizes the result of the opinion.

3. Approaches For Sentiment Analysis:

Sentiment Classification techniques can be categorized into machine learning approach, lexicon based approach and hybrid approach. In [5] Duwairi et.al states that polarity of the text is performed either using machine approach or lexicon based approach.

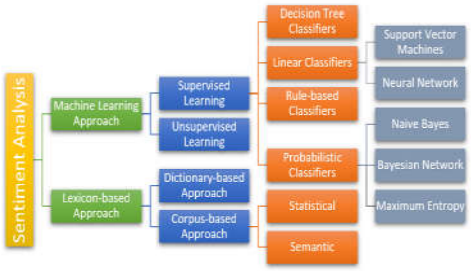

Figure 2: Sentiment Analysis Approaches – Souce: Devopedia

3.1. Machine Learning Approach:

It is a combination of both supervised learning and unsupervised learning. In supervised learning mainly focuses on classification of data, while unsupervised focuses on clustering. Mostly the opinions are classified as positive, negative or neutral. On

the basis of data classification machine learning approach applies algorithms and linguistic features.

3.2. Lexicon-based Approach:

The Lexicon-based approach depends on sentiment lexicons, an important tool for identifying the polarity of the texts and words. It is further classified into dictionary-based method (is a computational approach to measure the sensitivity of a text convey to the person who reads the text. In the simplest case, sentiment has a twofold sorting: positive or negative, but it can be extended to numerous extents such as panic, grief, irritation, joyfulness and so on. This method relies greatly on a pre-defined list (or dictionary) of sentiment-laden words. And corpus-based method (Helps to solve the problem of finding opinion words with context specific.).

3.3 Hybrid Approach:

It is the combination of both machine learning approach and lexicon based approach.

4. Natural Language Processing: Using TextBlob [6].

Natural Language Processing (NLP) mainly teaches the machine how to understand human language and extract meaning/essence from the transcript /speech.

Language acts as the medium of communication, which separates man and animal as we are floated with verbal and non-verbal data continually. Exciting NLP applications in real-life industry such as,

- Language Translation
- Dialog Systems / Chat bots
- Sentiment Analysis
- Text Summarizers
- Speech Recognition
- Autocorrect

My journey with NLTK library in Python. Modules used TextBlob and Emoji. Python needs to be installed in the python modules in Anaconda command prompt. Advantage is, it is easy to learn and offers a lot of features like sentiment analysis, pos-tagging and so on.

To install TextBlob – open anaconda prompts – and enter the following command:

```
pip install textblob
```

Installation of Emoji – enter the command:

```
pip install Emoji
```

Installation of NaiveBayesClassifier – enter the command:

```
pip install NaiveBayesClassifier
```

This will install the textblob, Emoji, and NaiveBayesClassifier modules.

For the untrained/uninitiated – practical work in Natural Language Processing typically uses large bodies of linguistic data, or **corpora.**

To download the necessary corpora, run the following command:

```
python –m textblob.download_corpora
```

TextBlob is a python library and offers a simple API to access its methods and perform basic NLP tasks.

Using TextBlob module I detected the polarity of a sentence and used Emoji to express the emotions.

```
from textblob import TextBlob, Word, Blobber
from textblob.classifiers import NaiveBayesClassifier
from textblob.taggers import NLTKTagger
import emoji
y=input("Enter the sentence:")
edu=TextBlob(y)

#using textblob.sentiment method

x=edu.sentiment.polarity
if x<0:
    print("Negative Sentence,demotivating",emoji.emojize(":disappointed_face:"))

elif x==0:
    print("Natural",emoji.emojize(":zipper-mouth_face:"))

elif x>0 and x<=1:
    print("Positive,Joyfull",emoji.emojize(":grinning_face_with_big_eyes:"))
```

```
Enter the sentence:im happy to see you
Positive,Joyfull 😃
```

```
from textblob import TextBlob, Word, Blobber
from textblob.classifiers import NaiveBayesClassifier
from textblob.taggers import NLTKTagger
import emoji
y=input("Enter the sentence:")
edu=TextBlob(y)

#using textblob.sentiment method

x=edu.sentiment.polarity
if x<0:
    print("Negative Sentence,demotivating",emoji.emojize(":disappointed_face:"))

elif x==0:
    print("Natural",emoji.emojize(":zipper-mouth_face:"))

elif x>0 and x<=1:
    print("Positive,Joyfull",emoji.emojize(":grinning_face_with_big_eyes:"))
```

```
Enter the sentence:im sad
Negative Sentence,demotivating 😨
```

```
from textblob import TextBlob, Word, Blobber
from textblob.classifiers import NaiveBayesClassifier
from textblob.taggers import NLTKTagger
import emoji
y=input("Enter the sentence:")
edu=TextBlob(y)

#using textblob.sentiment method

x=edu.sentiment.polarity
if x<0:
    print("Negative Sentence,demotivating",emoji.emojize(":disappointed_face:"))

elif x==0:
    print("Neutral,Natural",emoji.emojize(":zipper-mouth_face:"))

elif x>0 and x<=1:
    print("Positive,Joyfull",emoji.emojize(":grinning_face_with_big_eyes:"))
```

```
Enter the sentence:sun rises in the east
Neutral,Natural 😐
```

5. Sentiment Analysis

Sentiment analysis is mainly the procedure of shaping the attitude/mind-set or the sentiment/emotion of the writer, i.e., whether it is positive, negative or neutral.

The *sentiment* function of textblob returns two properties: **polarity** and **subjectivity**.

Polarity is float which lies in the range of [-1,1] where 1 means positive statement and -1 means a negative statement. Subjective sentences generally refer to personal opinion, emotion or /decision judgment whereas objective refers to factual information. Subjectivity is also a float which lies in the range of [0,1].

Let's check the sentiment of our blob.

```
print (blob)

blob. sentiment

>> Analytics Vidhya is a great platform to learn data science.

Sentiment (polarity=0.8, subjectivity=0.75)
```

```
print(blob)
blob.sentiment
```

```
Analytics Vidhya is a great platform to learn data science.
Sentiment(polarity=0.8, subjectivity=0.75)
```

We can see that polarity is **0.8**, which means that the statement is positive and **0.75** subjectivity refers that mostly it is a public opinion and not factual information.

6. Other ways to work with TextBlob:

6.1 Spelling Correction

Spelling correction is a cool feature which TextBlob offers; which can be accessed using the ***correct*** function as shown below.

```
blob = TextBlob('Analytics Vidhya is a gret platfrm to learn data scence')

blob.correct( )

>> TextBlob("Analytics Vidhya is a great platform to learn data science")
```

```
blob = TextBlob('Analytics Vidhya is a gret platfrm to learn data scence')
blob.correct( )
```

```
TextBlob("Analytics Vidhya is a great platform to learn data science")
```

We can also check the list of suggested word and its confidence using the ***spell check*** function.

```
blob.words[4].spellcheck( )

>> [('great', 0.5351351351351351),

 ('get', 0.3162162162162162),
```

```
('grew', 0.11216216216216217),
('grey', 0.026351351351351353),
('greet', 0.006081081081081081),
('fret', 0.00270270270270703),
('grit', 0.0006756756756756757),
('cret', 0.0006756756756756757)]
```

```
blob.words[4].spellcheck()
```

```
[('great', 0.5351351351351351),
 ('get', 0.3162162162162162),
 ('grew', 0.11216216216216217),
 ('grey', 0.026351351351351353),
 ('greet', 0.006081081081081081),
 ('fret', 0.00270270270270703),
 ('grit', 0.0006756756756756757),
 ('cret', 0.0006756756756756757)]
```

6.2 Creating a short summary of a text

This is a simple trick which we will be using the things we learned above

```
import random

blob = TextBlob('Analytics Vidhya is a thriving community for data driven industry. This platform allows \people to know more about analytics from its articles, Q&A forum, and learning paths. Also, we help \professionals & amateurs to sharpen their skillsets by providing a platform to participate in Hackathons.')
```

```
nouns = list( )
for word, tag in blob.tags:
if tag == 'NN':
nouns.append(word.lemmatize( ))

print ("This text is about...")
for item in random.sample(nouns, 5):
word = Word(item)
print (word.pluralize( ))
```

```
>> This text is about...
communities
platforms
forums
platforms
industries
```

```
import random
blob = TextBlob('Analytics Vidhya is a thriving community for data driven industry. This platform allows \people to know more abo
nouns = list( )
for word, tag in blob.tags:
    if tag == 'NN':
        nouns.append(word.lemmatize( ))

print ("This text is about...")
for item in random.sample(nouns, 5):
    word = Word(item)
print (word.pluralize( ))
```

```
This text is about...
communities
```

In the above code I extracted out a list of nouns from the text to give a general idea to the booklover about the things the data/manuscript is related to.

6.3 Translation and Language Detection

Guess what is written in the next line?

هذا بارد

Can you guess which language is this? We can't so, let's detect it using textblob…

```
blob.detect_language( )
```

```
>> 'ar'
```

```
from textblob import TextBlob
from langdetect import detect


L = ["Honesty is the best policy",
    "good morning - صباح الخير",
    ]

for i in L:

    # Language Detection
    blob = TextBlob(i)
    print(blob.detect_language())
```

```
en
ar
```

So, it is Arabic. Now, let's translate it into English so that we can know what is written in Arabic - using TextBlob.

```
blob.translate(from_lang='ar', to ='en')


>> TextBlob("that's cool")
```

```
blob.translate(from_lang='ar', to ='en')
```

```
TextBlob("good morning - good morning.")
```

Even if you don't explicitly define the source language, TextBlob will automatically detect the language and translate into the desired language.

```
blob.translate(to= 'en') ## or you can directly do like this


>> TextBlob("that's cool")
```

This is seriously so cool!!! □

```
from textblob import TextBlob
from langdetect import detect
L = ["صباح الخير"]
blob.translate(to= 'en')
```

```
TextBlob("good morning - good morning.")
```

7. Text classification using TextBlob

Let's build a simple text classification model using TextBlob. For this, first, we need to prepare training and testing data.

```
training = [
('Tom Holland is a terrible spiderman.','pos'),

('a terrible Javert (Russell Crowe) ruined Les Miserables for me...','pos'),

('The Dark Knight Rises is the greatest superhero moie ever!','neg'),

('Fantastic Four should have never been made.','pos'),

('Wes Anderson is my favorite director!','neg'),

('Captain America 2 is pretty awesome.','neg'),

('Let\s pretend "Batman and Robin" never happened..','pos'),

]


testing = [

('Superman was never an interesting character.','pos'),

('Fantastic Mr Fox is an awesome film!','neg'),

('Dragonball Evolution is simply terrible!!','pos')

]
```

Textblob provides in-built classifiers module to create a custom classifier. So, import it and create a basic classifier.

```
from textblob import classifiers

classifier =

classifiers. NaiveBayesClassifier (training)
```

We have passed the training data into the classifier.

Note: Here we have used Naive Bayes classifier, but TextBlob also offers Decision tree classifier which is as shown below.

```
## decision tree classifier

dt_classifier =

classifiers.DecisionTreeClassifier(training)
```

Now, let's check the accuracy of this classifier on the testing dataset and also TextBlob provides us to check the most informative features.

```
print (classifier. accuracy(testing))

classifier.show_informative_features (3)

>> 1.0


Most Informative Features


contains(is) = True    neg : pos    =    2.9 : 1.0

contains(terrible) = False neg : pos  =  1.8 : 1.0

contains(never) = False  neg : pos    =  1.8 : 1.0
```

```
training = [
('Tom Holland is a terrible spiderman.','pos'),

('a terrible Javert (Russell Crowe) ruined Les Miserables for me...','pos'),
('The Dark Knight Rises is the greatest superhero moie ever!','neg'),
('Fantastic Four should have never been made.','pos'),
('Wes Anderson is my favorite director!','neg'),
('Captain America 2 is pretty awesome.','neg'),
('Let\s pretend "Batman and Robin" never happened..','pos'),
]
testing = [
('Superman was never an interesting character.','pos'),
('Fantastic Mr Fox is an awesome film!','neg'),
('Dragonball Evolution is simply terrible!!','pos')
]
from textblcb import classifiers
classifier = classifiers. NaiveBayesClassifier (training)

## decision tree classifier
dt_classifier = classifiers.DecisionTreeClassifier(training)
print (classifier.accuracy(testing))
classifier.show_informative_features (3)
```

```
1.0
Most Informative Features
          contains(is) = True              neg : pos    =      2.9 : 1.0
           contains(a) = False             neg : pos    =      1.8 : 1.0
        contains(never) = False            neg : pos    =      1.8 : 1.0
```

If the text contains "is", then there is a high possibility that the statement will be negative.

To get some more idea, let's check our classifier on a random text.

```
blob = TextBlob('the weather is terrible!',

classifier=classifier)

print (blob.classify())


>> neg
```

```
blob = TextBlob('the weather is terrible!', classifier=classifier)
print (blob.classify())
```

```
neg
```

So, based on the training on the above dataset, our classifier has provided us the right result.

Note that here we could have done some preprocessing and data cleaning but here my aim is to give you an intuition that how we can do text classification using TextBlob.

8. Conclusion:

Sentiment analysis helps in identifying people's emotional and attitude state. People's sensitivity can be expressed in positive, negative and neutral way. Sentiment analysis can be extremely compelling in foreseeing assessment about stock exchanges or motion picture reviews like Imdb audits of face book and twitter can be likewise used to give useful information which can be utilized to think likely in future.

9. Acknowledgements:

I am thankful to Mr.Avinash A faculty of Centurion University of Technology and Management, Visakhapatnam, Andhra Pradesh, India for his cooperation and suggestions

References:

[1] "Current State of Text Sentiment Analysis from Opinion to Emotion Mining". ALI YADOLLAHI, AMENEH GHOLIPOUR SHAHRAKI, and OSMAR R. ZAIANE. ACM Computing Surveys (CSUR) 50, no.2 (2017):25.

[2] "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews." In proceedings of the 40th annual meeting on association for associational linguistics, pp.417-424.Association for Computational Linguistics, 2002.

[3] "Thumbs up? Sentiment Classification using Machine Learning Techniques". Proceedings of the Conference on Empirical Methods in Natural. Language Processing (EMNLP), Philadelphia, July 2002, pp. 79-86. Association for Computational Linguistics.

[4] Liu, Bing. "Many Facets of Sentiment Analysis." In A practical guide of Sentiment analysis, pp11-39. Springer International Publishing, 2017.

[5] "A Survey of Arabic Text Mining". Salloum, S.A., AlHamad, A, Q., Al-Emran, M., & Shaalan, K (2018). In Intelligent Natural Language Processing: Trends and Applications (pp.417-431).Springer. Cham.

[6] https://www.analyticsvidhya.com/blog/2018/02/natural-language-processing-for-beginners-using-textblob/