

SENTIMENT ANALYSIS ON FORMAL AND INFORMAL LANGUAGE

A SENIOR DESIGN PROJECT REPORT

*Submitted in partial fulfillment of the
requirement for the award of the
Degree of*

**BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE ENGINEERING WITH SPECIALIZATION IN DATA
ANALYTICS**

by

Lakshya Kumar(17BCD7037)

Under the Guidance of

DR. Karthika Natarajan



**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
VIT-AP UNIVERSITY
AMARAVATI- 522237**

JUNE 2022

SENTIMENT ANALYSIS ON FORMAL AND INFORMAL LANGUAGE

A SENIOR DESIGN PROJECT REPORT

*Submitted in partial fulfillment of the
requirement for the award of the
Degree of*

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE ENGINEERING WITH SPECIALIZATION IN DATA ANALYTICS

by

Lakshya Kumar(17BCD7037)

Under the Guidance of

DR. Karthika Natarajan



**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
VIT-AP UNIVERSITY
AMARAVATI- 522237**


JUNE 2022

DECLARATION

I here by declare that the thesis entitled “ SMART CROP PREDICTION AND MONITORING SYSTEM” submitted by me, for the award of the degree of Bachelor of Computer Science Engineering with Specialization in Data Analytics VIT is a record of bonafide work carried out by me under the supervision of Dr. Karthika Natarajan.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Amaravati
Date: Jun 12th, 2022
Candidate

A handwritten signature in black ink, appearing to read 'J akshya', written over a horizontal line.

Signature of the

CERTIFICATE

This is to certify that the Senior Design Project work titled “**SENTIMENT ANALYSIS ON FORMAL AND INFORMAL LANGUAGE**” that is being submitted by **LAKSHYA KUMAR(17BCD7037)** is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

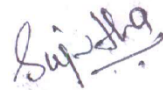


Dr. Karthika Natarajan
Guide

The thesis is satisfactory



Internal Examiner



Internal Examiner

Approved by



PROGRAM CHAIR

B. Tech. CSE-DA



DEAN

School of Computer Science and Engineering

B. Tech. CSE
Engineering

School of Computer Science and

ACKNOWLEDGEMENTS

It is my pleasure to express with deep sense of gratitude to Dr. Karthika Natarajan, Assistant Professor, School of Computer Science and Engineering, VIT-AP, for her constant guidance, continual encouragement, understanding; more than all, she taught me patience in my endeavor. My association with her is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Data Analytics.

I would like to express my gratitude to Dr. G. Viswanathan, Sankar Viswanathan, Dr. Sekar Viswanathan, G. V. Selvam, Dr. S. V. Kota Reddy, and Dr. Sudha S V, School of computer science and engineering, for providing with an environment to work in and for his inspiration during the tenure of the course.

In jubilant mood I express ingeniously my whole-hearted thanks to Dr. Mehfooza.M. Program Chair CSE DA, all teaching staff and members working as limbs of our university for their not-self-centered enthusiasm coupled with timely encouragements showered on me with zeal, which prompted the acquirement of the requisite knowledge to finalize my course study successfully. I would like to thank my parents for their support.

It is indeed a pleasure to thank my friends who persuaded and encouraged me to take up and complete this task. At last but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Amaravati
Kumar

Lakshya

Date: June 12, 2022

Name of the student

ABSTRACT

In this project an efficient and accurate model is presented which can analyze and quantify the sentiment behind statements on large scale datasets. Sentiment analysis is an ongoing field of research in text mining. Sentiment analysis is the computational treatment of opinions, sentiments and subjectivity of text. This project involves the study of several deep learning sentiment analysis models, their applications, their challenges, and construction of a sentiment analysis model of one of the proposed approach that best suits our needs (Classification of sentiments in Twitter and Metacritic data). In this project I aim to use the Valence Aware Dictionary for sentiment Reasoner (VADER) to classify the sentiments expressed in Twitter data and Metacritic data.

TABLE OF CONTENTS

S.No.	Chapter	Title	Page Number
1.		Acknowledgement	3
2.		Abstract	4
3.		List of Figures and Table	6
4.	1	Introduction	7
	1.1	Objectives	7
	1.2	Background and Literature Survey	9
	1.3	Organization of the Report	12
5.	2	SENTIMENT ANALYSIS ON FORMAL AND INFORMAL LANGUAGE	13
	2.1	Proposed System	13
	2.2	Working Methodology	14
	2.3	System Details	15
	2.3.1	Software	15
6.	3	Cost Analysis	22
	3.1	List of subscription and their cost	22
7.	4	Results and Discussion	23
8.	5	Conclusion & Future Works	24
9.	6	Appendix	25
10.	7	References	32

List of Tables

Table No.	Title	Page No.
1.	Cost Analysis	37

List of Figures

Figure No.	Title	Page No.
1	Workflow for Sentiment Analysis	13
2	Workflow for aspect-based sentiment analysis	14
3	Description of twitter dataset	15
4	Use of the isnull function	16
5	Description of Metacritic dataset	16
6	Use of the isnull function	17
7	Libraries imported	18
8	Implementation of text cleaning	19
9	Definition of get_wordnet_pos function	19
10	Function to find the sentiment label	20

CHAPTER 1

INTRODUCTION

Natural language processing (NLP) is a field of artificial intelligence in which computers analyze, understand, and derive meaning from human language in a smart and useful way. By utilizing NLP, developers can organize and structure knowledge to perform tasks such as automatic summarization, translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition, and topic segmentation.

Sentiment analysis is an approach to natural language processing which identifies the emotional tone behind a body of text.

In order to study sentiment analysis, we first have to ask ourselves some basic questions. What is sentiment analysis? Why do we need it? What are its applications? What are the different types of sentiment analysis? and how many ways are there to perform these tasks? We will uncover these topics in our project.

1.1 Objectives

The following are the objectives of this project:

To perform Sentiment Analysis on brief text statements acquired from product evaluations and tweets using Natural Language Processing (NLP).

Following the generation of overall sentiment analysis for the statements, the goal is to determine the subject in the statement.

The project's subsequent study is to determine if the same reasoning can be applied in statements including informal material (i.e., language used on social media).

For the formal and informal structures we shall use Metacritic Game Reviews and Tweets respectively as they capture the essence of the structures we aim to test.

1.2 Background and Literature Survey

Sentiment analysis is a very well documented field, so in order to understand and study the field I started with the basic question, what is sentiment analysis?

Sentiment analysis is the study of people's sentiments, opinions, feedback, attitude towards a certain product or services and their attributes. With the advent of the internet age large amounts of data is collected via our search histories, tweets, Instagram posts, the internet generates around 2.5 quintillion bytes of data every day. Which makes sentiment analysis one of the largest studied field in deep learning and data mining. It can analyze everything from the direction in which people are going to vote in the next elections to which movie is bound to be the next block-buster. Nowadays if you want to buy a new phone, instead of asking a family member or a friend for recommendation you go to google reviews to search for the products reviews and ratings, on an individual level one review means nothing to a company but collectively on a global scale these small reviews can impact a company in millions of dollars. But the reviews or tweets on the internet may contain emojis or incorrect grammar, or even sarcasm. This makes the task especially challenging to look through these layers of language and find out the essence of any message on the internet.

Now the question arises what are different types of sentiment analysis? To answer this question I referred a paper by Ronen Feldman named "Techniques and Applications for Sentiment Analysis". In which he describes that sentiment analysis can be of three types document-level sentiment analysis, sentence-level sentiment analysis and aspect level sentiment analysis.

Document level sentiment analysis is the case when an author expresses one opinion about the subject matter in a document. This task can be done using supervised machine learning models i.e. the number of classes in which the document is supposed to be classified is finite (positive, negative or neutral). We can also use a five star rating system like amazon. Algorithms like SVM, Naive Bayes or KNN can be used for these tasks. This is one of the easiest analysis that can be done. We can also approach this topic through unsupervised learning in which we evaluate

the semantic orientation of specific phrases in the document and if the sum of their values crosses a certain threshold it is classified as positive or negative.

Sentence-level sentiment analysis: A single document may contain multiple views about the same entity, for example the chilly cheese fries were salty but also crisp and delicious. In the example a small sentence contains two different views so a document will contain multiple views, in order to tackle that sentence level sentiment analysis is introduced.

Aspect-Base sentiment analysis the previous two approaches work when the document or the sentence talks about one attribute. But what happens when in a discussion forum about cars people talks about different cars, their particular engines, the horsepower or comfort of the car. In such cases when the attributes are multiple aspect-based sentiment analysis works best.

Upon further inspection of related literature, I found that most commonly there are three ways to perform the above mentioned tasks, they are Naive Bayes, Deep Learning LSTM, and VADER Model. In order to find the most efficient way I studied these three models in detail.

Naive Bayes approach: In this the Naive Bayes classifier utilizes the probabilities based on prior knowledge of the conditions that might be related. It does that by construction of a document term matrix for the model. This approach is easy to understand and fairly reasonable. The use of this classifier requires the preprocessing of data. For capturing the context of words N-grams are used , but N-grams being blunt instruments can't always capture the essence correctly and they can burden the model with too many features. The data preprocessing includes the standard steps(Converting case, tokenize, lemmatize, stop word removals) with an addition of vectorizing the words(TF-IDF).

Deep Learning LSTM approach: LSTM stands for Long-Short Term Memory, is a type of RNN used to process temporal data (data relating to time related instances). Deep learning is computationally expensive and does not do well high dimensional sparse vectors. In this technique each word is converted into a sequence of numbers where each number is mapped to a word in the vocabulary, Taking it a step further,

we need to map words that have similar usage/meaning to similar real number vectors (rather than an index) using Word Embeddings. The preprocessing follows the standard process mentioned in naive bayes, and while vectorizing the text is converted into sequence and word embedding is used, this step comes under the vectorizing of words. The big advantage of this model as compared to naive bayes is that we don't have to engineer features, since the model will automatically learn the features during training making the model highly accurate.

VADER model: *“VADER (Valence Aware Dictionary and sentiment Reasoner) tool. At its core, VADER uses a comprehensive, high quality lexicon (~7500 features) and sophisticated linguistic rules to produce sentiment scores”*. Vader being a valance based lexicon, is capable of detecting both the intensity and polarity of sentiments. This tool combined with modifiers such as negations, contractions, conjunctions etc. are used to calculate scores for sentences. The advantage of Vader over the other two models is the ability of the model to detect slangs and sarcasm while being highly efficient and fast. The real challenge for creating tools like this is the large amount of time it takes to create tools like Vader. The preprocessing for Vader follows, Converting case, tokenize, lemmatize, stop word removals followed by POS tagging.

Therefore picking the right tool will depend, if speed is prioritized over accuracy Vader should be used, if the accuracy is paramount Deep learning model should be used and if the training data is robust and accurate then Naive Bayes model should be used. In our case the data we are using is twitter and Metacritic review data which isn't robust and contains slangs and informal sentence structure. Hence we will be using Vader for our project.

In each of the models data preprocessing is the common step with some small changes, so data preprocessing in context of sentiment analysis was studied to determine what challenges it might entail. A methodology was determined after study of the Vader model and will be mentioned later in the report.

1.3 Organization of the Report

The remaining chapters of the project report are described as follows:

- Chapter 2 contains the proposed system, methodology, hardware and software details.
- Chapter 3 gives the cost involved in the implementation of the project.
- Chapter 4 discusses the results obtained after the project was implemented.
- Chapter 5 concludes the report.
- Chapter 6 consists of codes.
- Chapter 7 gives references.

CHAPTER 2

SENTIMENT ANALYSIS ON FORMAL AND INFORMAL LANGUAGE

This Chapter describes the proposed system, working methodology, software and hardware details.

2.1 Proposed System

Using NLTK, we performed sentiment analysis on the Metacritic Video Game Reviews. Using NLTK, we tokenized the statements and down the words to its simpler form (also referred to as cleaning of the text). This helps in figuring out the Parts-of-speech (POS) tags for the words in each statement, which is used by the model to perform sentiment analysis.

The model used is derived from the NLTK package in the Vader Module. The model is pretrained by the developers and any user must only clean and normalize the text before providing it as input to the model.

We then, performed sentiment analysis and classify them to three possible labels:

1. Positive
2. Neutral
3. Negative

The following diagram (figure 1) shows the algorithms used for sentiment analysis.

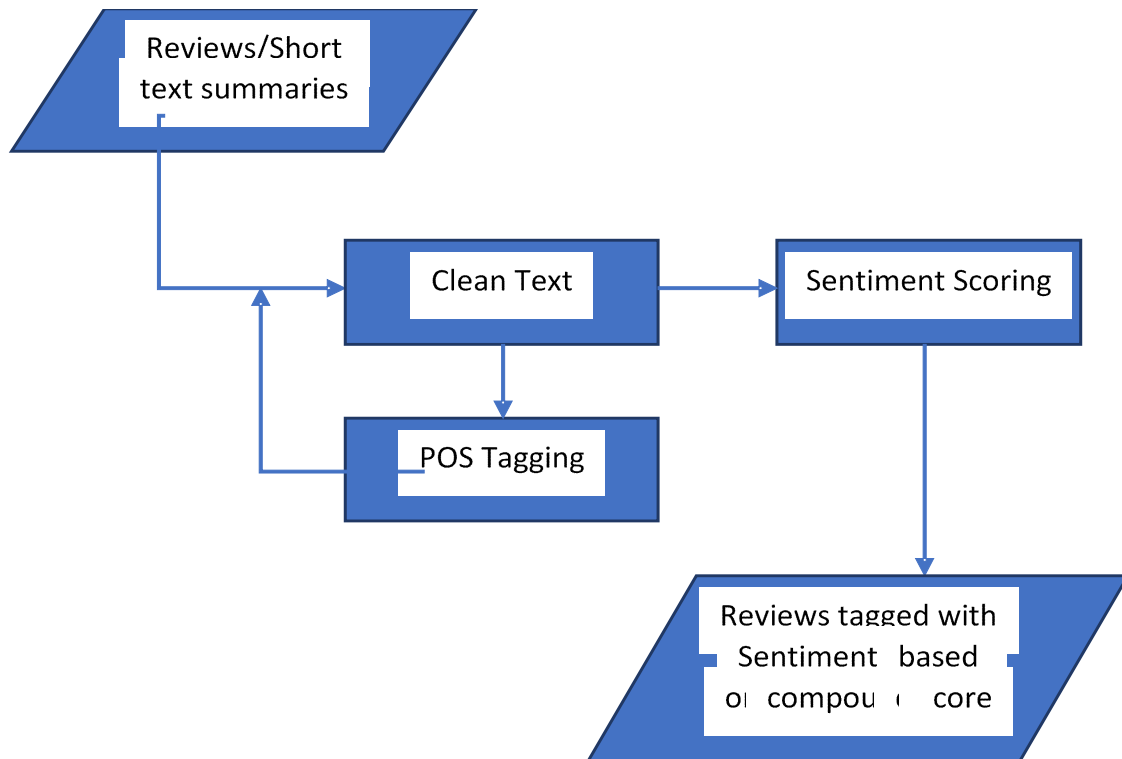


Figure 1 Workflow for Sentiment Analysis

For identifying the aspects, i.e., the parts that the sentiments are expressed towards. We look at the grammatical structure of the statement using spaCy, we can see that the aspects in all of the statements is identified as Nouns and the describing the nouns are Adjectives.

spaCy can parse and tag any statement after tokenization using its trained pipeline and its statistical models to forecast which tag or label is most likely to apply in this situation.

Upon the identification of the aspects and its descriptive word(s), we perform sentiment analysis to see the sentiments associated with the descriptive words.

The diagram (figure 2) shows the proposed workflow for aspect based sentiment analysis.

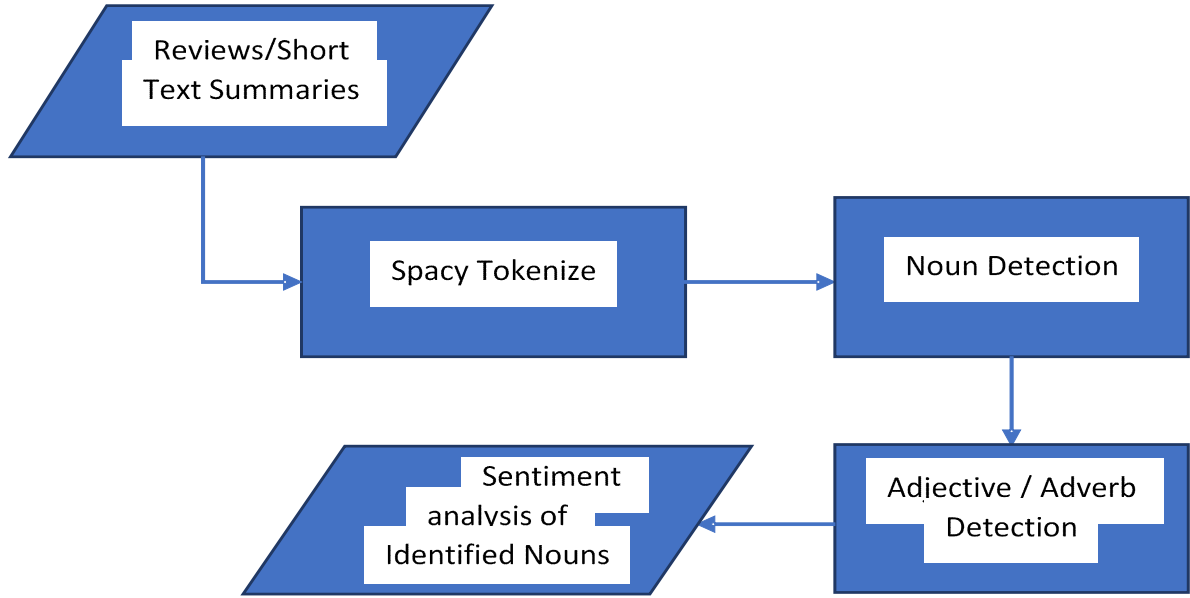


Figure 2 Workflow for aspect-based sentiment analysis

2.2 Working Methodology

This project has two parts:

Sentiment analysis, in which we calculate the polarity of sentences.

Aspect-based sentiment analysis, in which we are identifying the aspects, i.e., the parts that the sentiments are expressed towards.

Each of these tasks will be carried out on formal sentence structure(Metacritic Game Reviews), and on informal sentence structures(Tweets).

2.3 System Details

This section describes the software details of the system:

2.3.1 Software Details

First an analysis on both the datasets will be performed. This will provide us with a general overview of the dataset. We perform this using pandas library in python.

We start by importing the twitter dataset and displaying the first four rows of the dataset

```
In [16]: twitter = pd.read_csv("tweets.csv")
twitter.columns = twitter.columns.str.replace(' ', '')

In [17]: twitter.head()
Out[17]:
```

	id	review
0	31963	#studiolife #aislife #requires #passion #dedic...
1	31964	@user #white #supremacists want everyone to s...
2	31965	safe ways to heal your #acne!! #altwaystohe...
3	31966	is the hp and the cursed child book up for res...
4	31967	3rd #bñday to my amazing, hilarious #nephew...

```
In [18]: twitter.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17197 entries, 0 to 17196
Data columns (total 2 columns):
#   Column  Non-Null Count  Dtype
---  -
0    id      17197 non-null    int64
1   review  17197 non-null    object
dtypes: int64(1), object(1)
memory usage: 268.8+ KB
```

Figure 3 Description of twitter dataset

Now we run the describe function and isnull function in pandas to check for any null values or inconsistencies in the dataset. This will allow us to clean the data better during data preprocessing.

```
In [19]: twitter.describe()
Out[19]:
```

	id
count	17197.000000
mean	40561.000000
std	4964.490625
min	31963.000000
25%	36262.000000
50%	40561.000000
75%	44860.000000
max	49159.000000

```
In [20]: twitter.isnull().sum()
Out[20]: id      0
review  0
dtype: int64
```

Figure 4 Use of the isnull function

Now we follow the same steps for Metacritic.csv file

```
In [21]: # Metacritics
```

```
In [22]: meta = pd.read_csv("metacritic_critic_reviews.csv")
meta.columns = meta.columns.str.replace(' ', '')
```

```
In [23]: meta.head()
```

```
Out[23]:
```

	name	review	game	platform	score	date
0	LEVEL (Czech Republic)	Portal 2 is a masterpiece, a work of art that ...	Portal 2	PC	100.0	25-May-11
1	GameCritics	So do we need Portal 2? Do I need it? Maybe no...	Portal 2	PC	100.0	8-May-11
2	PC Games (Russia)	Portal 2 exceeds every expectation. It has a s...	Portal 2	PC	100.0	6-May-11
3	Adventure Gamers	Like its predecessor, Portal 2 is not an adven...	Portal 2	PC	100.0	29-Apr-11
4	Armchair Empire	Pile on the "Oh, yes!" moments of solving some...	Portal 2	PC	100.0	28-Apr-11

```
In [24]: meta.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125876 entries, 0 to 125875
Data columns (total 6 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   name        125876 non-null object
1   review      125876 non-null object
2   game        125876 non-null object
3   platform    125876 non-null object
4   score       124311 non-null float64
5   date        125832 non-null object
dtypes: float64(1), object(5)
memory usage: 5.8+ MB
```

Figure 5 Description of Metacritic dataset

```
In [25]: meta.describe()
```

```
Out[25]:
```

	score
count	124311.000000
mean	74.495161
std	14.511340
min	0.000000
25%	70.000000
50%	80.000000
75%	85.000000
max	100.000000

```
In [26]: meta.isnull().sum()
```

```
Out[26]: name        0
review      0
game        0
platform    0
score       1565
date        44
dtype: int64
```

Figure 6 Use of the isnull function

Now that we have a general idea of what is in our dataset we shall start with data preprocessing for sentiment analysis and aspect-based sentiment analysis.

We start by importing libraries

```
1 import pandas as pd # Pandas is used for dataset operation
2 import string # String is used for identifying parts of grammar; in the program's case it is punctuations
3 from nltk import pos_tag # Identifies the Parts-of-Speech tags of the word, like, "Canada" is noun
4 from nltk.corpus import stopwords # A list of words that don't add a significant meaning to the statement during processing
5 from nltk.stem import WordNetLemmatizer # Lemmatization is a process which converts words in multiple forms to a easy to
6                                     # process word, like, "runs" and "running" turns to "run"
7 from nltk.corpus import wordnet
8 from nltk.sentiment.vader import SentimentIntensityAnalyzer # The Sentiment Analysis model which is trained and is readily available by the modulw developer
9 from progress.bar import Bar # Adding a progress bar to keep track of the progress of the functions
10 import spacy # A superior form of the NLTK package which offers better analytical tools for examining the relationships between the words
11
12 print("Packages Loaded\n")
--
```

Figure 7 Libraries imported

1. Pandas is used for dataset operations
2. String is used for identifying parts of grammar; in the program's case it is punctuation
3. pos_tag, Identifies the Parts-of-Speech tags of the word, like, "Canada" is noun
4. stopwords is a list of words that don't add a significant meaning to the statement during processing
5. WordNetLemmatizer, Lemmatization is a process which converts words in multiple forms to a easy to process word, like, "runs" and "running" turns to "run"
6. SentimentIntensityAnalyzer : The Sentiment Analysis model which is trained and is readily available by the module developers
7. Bar : Adding a progress bar to keep track of the progress of the functions
8. spacy : A superior form of the NLTK package which offers better analytical tools for examining the relationships between the words

Now we define a function to clean the text that preprocesses the data by performing the following steps

1. converts the text into lowercase
2. tokenize the text and remove all punctuation
3. Remove words that contain numbers
4. Remove stop words
5. Remove empty tokens
6. POS tag the text
7. Lemmatize the text

8. Remove words with only one letter

The code will look like the image below.

```
# The function below cleans the text, so that the Sentiment Analysis model can process the information effectively
def clean_text(text):
    # Lower text
    text = text.lower()
    # Tokenize text and remove punctuation
    text = [word.strip(string.punctuation) for word in text.split(" ")]
    # Remove words that contain numbers
    text = [word for word in text if not any(c.isdigit() for c in word)]
    # Remove stop words
    stop = stopwords.words('english')
    text = [x for x in text if x not in stop]
    # Remove empty tokens
    text = [t for t in text if len(t) > 0]
    # POS tag text
    pos_tags = pos_tag(text)
    # Lemmatize text
    text = [WordNetLemmatizer().lemmatize(t[0], get_wordnet_pos(t[1])) for t in pos_tags]
    # Remove words with only one letter
    text = [t for t in text if len(t) > 1]
    # Join all
    text = " ".join(text)
    return(text)
```

Figure 8 Implementation of text cleaning

The function above use the function `get_wordnet_pos`, so we will define this function next. This function will convert the POS tag from a string form to the wordnet format. The implementation of the function is a simple if else structure that is shown in the figure below.

```
# The function below works with the clean_text function which will convert the POS tag from a string form to the wordnet format
def get_wordnet_pos(pos_tag):
    if pos_tag.startswith('J'):
        return wordnet.ADJ
    elif pos_tag.startswith('V'):
        return wordnet.VERB
    elif pos_tag.startswith('N'):
        return wordnet.NOUN
    elif pos_tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN
```

Figure 9 Definition of `get_wordnet_pos` function

The sentiment intensity analyzer function returns a number value so in order to convert the number into an easy readable label we will define the `sentiment_sen()` function, where

if the score is greater than 0 it returns the label positive, else if the score is less than -0.10 it returns the label negative, else it returns neutral.

```
# This function will convert the compound score generated by the Sentiment Analyser model to a easy to understand label
# More labels can be added based on the user's preference
# The limits set are arbitrary and can be modified based on the user's preference
def sentiment_sen(x):
    if x >= 0.00 :
        return("Positive")

    elif x <= - 0.10 :
        return("Negative")

    else :
        return("Neutral")
```

Figure 10 Function to find the sentiment label

Now that we have the functions defined we can start with the sentiment and aspect-based sentiment analysis.

Sentiment Analysis :

1. First import the dataset
2. Then select the review column
3. Initialize the progress bar to keep track of the progress.
4. Clean the text using the above function for each review
5. All the clean text is passed through the sentiment intensity analyzer to generate polarity scores.
6. Use the generated scores to generate labels for each review.

Aspect-based sentiment analysis:

1. Define a function that finds sentiments, similar to the code segment above, but are only applied for words or small sets of words to understand its sentiments
2. Each review is analyzed to identify the aspects, the words that provide its sentiments and the sentiments towards the aspects
3. Use spacy to tokenize each sentence
4. For each of the tokens look at the words in the statement and if they are nouns that have links (i.e 'nsubj') then they are the aspects, and if the words

are adjectives and are not linked to any adverbs then they are the descriptive terms

5. Add this information in a dictionary format containing the aspects, description and sentiments
6. Find sentiment for each of these aspects and add it to a new column in the csv file.
7. Save the csv file under a new name with its sentiment analysis and aspect-based sentiment analysis.

Repeat this process for Metacritic dataset.

CHAPTER 3

COST ANALYSIS

3.1 List of Subscription and their cost

The costs of the various journal subscriptions referred in this project are given below in Table 3.1.

Table 3.1 List of Subscription and their costs

COMPONENT	COST
IEEE Xplore subscription	₹ 3600
TOTAL	₹ 3600

CHAPTER 4

RESULTS AND DISCUSSIONS

A. Metacritic Game Reviews

The program is able to identify the overall sentiments of the results with an F-1 accuracy score of ~82.921%. From the findings, we can see that NLTK provides a fast and reliable method to identify the sentiments. It should also be reiterated that the limits of labeling the polarity scores along with the tagging of “True Values” for the F1 testing are set by the individual and its modification will lead to different accuracy scores.

The aspect-based sentiment analysis is able to successfully identify the certain aspects of the statement and also its sentiments. However, there are statements where the dependencies were not found but the sentiments were detected. Also, in some cases the detected nouns didn’t make any sense.

Another issue with the method proposed in the aspect-based sentiment analysis [3], is that the program can only detect one aspect, if a complex statement is provided with multiple nouns and is described with various adjectives then the program may not be able to detect the aspects properly.

For future implementation, the recommendations are to:

1. Modify the hyper-parameters to improve accuracy, precision, recall, etc.
2. Modify the code to detect multiple aspects
3. Adjust for the sarcastic tones

B. Twitter Tweets

The sentiments from the tweets are accurately captured. However, the dataset requires cleaning as it contains emoticons, which the program interprets as special characters and messes with the spaCy interpretation of the statement.

The CSV files containing complete outputs are attached with the report.

CHAPTER 5

CONCLUSION AND FUTURE WORK

For future implementation, the recommendations are to:

1. An automated data cleaning method to normalize all characters, especially emoticons
2. Conversion of abbreviations like WTH or TMI to a form that can be interpreted by the program

However, the above recommendations have a disadvantage of requiring a significant amount of time and resources to process the information, but the advantage is that a model will be used that is accurate and reliable.

Another possible method is to create a new custom model which operates on a similar logic and produces similar results but is tweaked for the inclusion for informal structures. This also brings an advantage of not preprocessing the information to a great extent, but the disadvantages are reworking the entire model which might affect its accuracy and will also require knowledge and time in terms of development.

CHAPTER 6

APPENDIX

Python Code For Sentiment and Aspect-based Sentiment Analysis

```
import pandas as pd # Pandas is used for dataset operation

import string # String is used for identifying parts of grammar; in the program's case it is
punctuations

from nltk import pos_tag # Identifies the Parts-of-Speech tags of the word, like, "Canada"
is noun

from nltk.corpus import stopwords # A list of words that don't add a significant meaning
to the statement during processing

from nltk.stem import WordNetLemmatizer # Lemmatization is a process which converts
words in multiple forms to a easy to

                                # process word, like, "runs" and "running" turns to "run"

from nltk.corpus import wordnet

from nltk.sentiment.vader import SentimentIntensityAnalyzer # The Sentiment Analysis
model which is trained and is readily available by the modulw developers

from progress.bar import Bar # Adding a progress bar to keep track of the progress of the
functions

import spacy # A superior form of the NLTK package which offers better analytical tools
for examining the relationships between the words

print("Packages Loaded\n")

# spaCy offers the computations operating on the GPU, which during development
showed the total time taken to process the data significantly reduce

# If the user doesn't have a GPU, please comment the statement below

spacy.prefer_gpu()

nlp = spacy.load("en_core_web_sm") # A pretrained module which can tokenize the
statements
```

The function below cleans the text, so that the Sentiment Analysis model can process the information effectively

```
def clean_text(text):  
    # Lower text  
    text = text.lower()  
  
    # Tokenize text and remove puncutation  
    text = [word.strip(string.punctuation) for word in text.split(" ")]  
  
    # Remove words that contain numbers  
    text = [word for word in text if not any(c.isdigit() for c in word)]  
  
    # Remove stop words  
    stop = stopwords.words('english')  
    text = [x for x in text if x not in stop]  
  
    # Remove empty tokens  
    text = [t for t in text if len(t) > 0]  
  
    # POS tag text  
    pos_tags = pos_tag(text)  
  
    # Lemmatize text  
    text = [WordNetLemmatizer().lemmatize(t[0], get_wordnet_pos(t[1])) for t in pos_tags]  
  
    # Remove words with only one letter  
    text = [t for t in text if len(t) > 1]  
  
    # Join all  
    text = " ".join(text)  
  
    return(text)
```

The function below works with the clean_text function which will convert the POS tag from a string form to the wordnet format

```
def get_wordnet_pos(pos_tag):
```

```

if pos_tag.startswith('J'):
    return wordnet.ADJ
elif pos_tag.startswith('V'):
    return wordnet.VERB
elif pos_tag.startswith('N'):
    return wordnet.NOUN
elif pos_tag.startswith('R'):
    return wordnet.ADV
else:
    return wordnet.NOUN

# This function will convert the compound score generated by the Sentiment Analyser
# model to a easy to understand label
# More labels can be added based on the user's preference
# The limits set are arbitrary and can be modified based on the user's preference
def sentiment_sen(x):
    if x >= 0.00 :
        return("Positive")

    elif x <= - 0.10 :
        return("Negative")

    else :
        return("Neutral")

# Load the dataset; dataset must be a CSV form and the user must change the file name to
# read any other file
# The statements that are to be processed must belong in the column 'review'
reviews = pd.read_csv("metacritic_critic_reviews.csv")

```

```

print("#####START#####")
print("\nFormat Cells")

rev = reviews['review'] # Select the review column only

print("\nNormalise Reviews")

bar = Bar('Processing', fill = "|", max=len(reviews), suffix='%(percent).1f%%') # Set up a
progress bar to track the progress

for i in range(len(reviews)):

    reviews.loc[i,'n_review'] = clean_text(reviews.loc[i,'review']) # Each statement is
cleaned and tokenized and are then ammended to a new column in the dataset

    bar.next()

bar.finish()

print("\nSentiment Analysis")

sid = SentimentIntensityAnalyzer()

# All clean/normalized text are provided as input to the model and the polarity socres are
generated

reviews["sentiments"] = reviews["n_review"].apply(lambda x: sid.polarity_scores(x))

reviews = pd.concat([reviews.drop(['sentiments'], axis=1),
reviews['sentiments'].apply(pd.Series)], axis=1) # Adjustments are made to make the
dataset more readable

reviews["sentiment"] = reviews["compound"].apply(lambda x: sentiment_sen(x)) # The
socres generated are converted to lables

print("#####ASPECT BASED SENTIMENT
ANALYSIS#####")

```

The following function finds sentiments, similar to the code segment above, but are only applied for words or small sets of words to understand it's sentiments

```
def find_sentiment(text):
```

```
    text = clean_text(text)
```

```
    sid = SentimentIntensityAnalyzer()
```

```
    sentiments = sid.polarity_scores(text)
```

```
    return sentiment_sen(sentiments['compound'])
```

```
aspects = [] # Aspects are the parts of the statement that the sentiments are reffering
```

```
sentences = reviews['review'].to_list()
```

```
print("Aspect Sentiments")
```

```
# Following couple lines of code are for the progress bar to keep track of the progress
```

```
length = len(sentences)
```

```
bar = Bar('Processing', fill = "|", max=length, suffix='%(percent).1f%%')
```

```
# Each review is analyzed to identify the aspects, the words that provide its sentiments  
and the sentiments towards the aspects
```

```
for i in range(length):
```

```
    sentence = sentences[i]
```

```
    doc = nlp(sentence)
```

```
    for token in doc:
```

```
        descriptive_term = "
```

```
        target = "
```

```
        # All sentiment delivering words are Adjectives and it describes Nouns
```

```
        # The following lines of code look at the words in the statement and if they are  
nouns that have links (i.e 'nsubj') then they are the aspects
```

```
        # And if the words are adjectives and are not linked to any adverbs then they are the  
descriptive terms
```

```
        if token.dep_ == 'nsubj' and token.pos_ == 'NOUN':
```

```
            target = token.text
```



```

        if token.pos_ == 'ADJ':
            prepend = ""
            for child in token.children:
                if child.pos_ != 'ADV':
                    continue
                prepend += child.text + ' '
            descriptive_term = prepend + token.text
            aspects.append({'aspect': target,
                           'description': descriptive_term,
                           'sentiment': find_sentiment(descriptive_term)}) # Adds the information in a
            # dictionary format containing the aspects, description and sentiment

    bar.next()
bar.finish

reviews['aspect_sentiment'] = pd.Series(aspects) # Converts to a pandas series and adds
it to the final resulting dataset

reviews.to_csv("reviews_sentiment.csv") # Saves the resulting dataset as a CSV file
print("\nReviews with overall sentiment analysis and aspect based sentiment analysis
saved as a .csv file")

print("#####FINISH#####")

```

Dataset analysis code

```

'''
    This notebook contains the python code for general analysis of twitter and metacritic
    datasets
'''

# Twitter dataset

import pandas as pd # data analysis

```

```

import numpy as np # linear algebra

#import libraries for data visualization
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

twitter = pd.read_csv("tweets.csv")
twitter.columns = twitter.columns.str.replace(' ', '')
twitter.head()

twitter.info()

twitter.describe()
twitter.isnull().sum()

# Metacritics

meta = pd.read_csv("metacritic_critic_reviews.csv")
meta.columns = meta.columns.str.replace(' ', '')
meta.head()
meta.info()
meta.describe()
meta.isnull().sum()

```

REFERENCES

1. Algorithmia. 11th August 2016. What is natural language processing? Introduction to NLP. [online] Available at: <https://algorithmia.com/blog/introduction-natural-language-processing-nlp>
2. GeeksForGeeks. Python | Sentiment Analysis using VADER. [online] Available At: <https://www.geeksforgeeks.org/python-sentiment-analysis-using-vader/>
3. Pattakos, A. (2021). Aspect-Based Sentiment Analysis Using Spacy & TextBlob. Towards Data Science. [online] Available at: <https://towardsdatascience.com/aspect-based-sentiment-analysis-using-spacy-textblob-4c8de3e0d2b9>
4. spaCy. Facts-Figures. [online] Available at: <https://spacy.io/usage/facts-figures>
5. spaCy. Linguistic Features. Available at: <https://spacy.io/usage/linguistic-features>
6. Doaa Mohey El-Din Mohamed Hussein, A survey on sentiment analysis challenges, Journal of King Saud University - Engineering Sciences, Volume 30, Issue 4, 2018, Pages 330-338, ISSN 1018-3639, <https://doi.org/10.1016/j.jksues.2016.04.002>. (<https://www.sciencedirect.com/science/article/pii/S1018363916300071>)
7. Sentiment Analysis and Opinion Mining, Bing Liu, Synthesis Lectures on Human Language Technologies, May 2012, Vol. 5, No. 1, Pages 1-167, (<https://doi.org/10.2200/S00416ED1V01Y201204HLT016>)
8. Emma Haddi, Xiaohui Liu, Yong Shi, The Role of Text Pre-processing in Sentiment Analysis, Procedia Computer Science, Volume 17, 2013, Pages 26-32, ISSN 1877-0509, <https://doi.org/10.1016/j.procs.2013.05.005>. (<https://www.sciencedirect.com/science/article/pii/S1877050913001385>)
9. <https://towardsdatascience.com/sentiment-analysis-comparing-3-common-approaches-naive-bayes-lstm-and-vader-ab561f834f89#:~:text=There%20are%20numerous%20approaches%20for,Trained%20Rule%20Based%20VADER%20Models>.
10. Zhang, Lei, Shuai Wang, and Bing Liu. "Deep learning for sentiment analysis: A survey." Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8.4 (2018): e1253.
11. Zhang, L., Wang, S. and Liu, B., 2018. Deep learning for sentiment analysis: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4), p.e1253.
12. Zhang, Lei, Shuai Wang, and Bing Liu. "Deep learning for sentiment analysis: A survey." Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8, no. 4 (2018): e1253.
13. Taboada, Maite & Brooke, Julian & Tofiloski, Milan & Voll, Kimberly & Stede, Manfred. (2011). Lexicon-Based Methods for Sentiment Analysis. Computational Linguistics. 37. 267-307. 10.1162/COLI_a_00049.
14. Agarwal, Apoorv, Boyi Xie, Ilia Vovsha, Owen Rambow and R. Passonneau. "Sentiment Analysis of Twitter Data." (2011).
15. Schouten, Kim & Frasincar, Flavius. (2015). Survey on Aspect-Level Sentiment Analysis. IEEE Transactions on Knowledge and Data Engineering. 28. 1-1. 10.1109/TKDE.2015.2485209.

16. Hutto, C.J. & Gilbert, Eric. (2015). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014.
17. Elbagir, Shihab & Yang, Jing. (2020). Sentiment Analysis on Twitter with Python's Natural Language Toolkit and VADER Sentiment Analyzer. 63-80. 10.1142/9789811215094_0005.
18. Zhang, Lei & Wang, Shuai & Liu, Bing. (2018). Deep Learning for Sentiment Analysis : A Survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 8. 10.1002/widm.1253.
19. Kiritchenko, Svetlana & Zhu, Xiaodan & Mohammad, Saif. (2014). Sentiment Analysis of Short Informal Text. The Journal of Artificial Intelligence Research (JAIR). 50. 10.1613/jair.4272.
20. Taboada, Maite. (2016). Sentiment Analysis: An Overview from Linguistics. Annual Review of Linguistics. 2. 10.1146/annurev-linguistics-011415-040518.
21. Hassan Yousef, Ahmed & Medhat, Walaa & Mohamed, Hoda. (2014). Sentiment Analysis Algorithms and Applications: A Survey. Ain Shams Engineering Journal. 5. 10.1016/j.asej.2014.04.011.
22. Feldman, Ronen. (2013). Techniques and Applications for Sentiment Analysis. Commun. ACM. 56. 82–89. 10.1145/2436256.2436274.
23. Mohey El-Din, Doaa. (2016). A Survey on Sentiment Analysis Challenges. Journal of King Saud University - Engineering Sciences. -. 10.1016/j.jksues.2016.04.002.

BIODATA



Name : Lakshya Kumar

Mobile Number : 9620792339

E-mail : lakshya.kumar@vitap.ac.in

Permanaent Address : 202 Sri Sai Nilayam, 7th cross, Chinnapanhalli Bangalore 560037