

**A Project/Dissertation ETE Report**

**On**

**SENTIMENT ANALYSIS  
USING TEXT FEEDBACK**

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

**Bachelor of Technology (CSE)**



**Under The Supervision of  
Mr. S Rakesh Kumar**

**Submitted By**

**Nishit Parashar(19SCSE1010404)  
Nikhil Kumar (19SCSE1010532)**

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING  
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
GALGOTIAS UNIVERSITY, GREATER NOIDA  
INDIA  
December , 2021**

## ACKNOWLEDGEMENT

We would like to express special thanks & gratitude to our guide, Mr.S Rakesh Kumar who gave us this golden opportunity to work on this scalable project on the topic of “**Sentiment Analysis from Text Feedback**”, which led us into doing a lot of Research which diversified our knowledge to a huge extent for which we are thankful.

-----  
NISHIT PARASHAR(19SCSE1010404)

-----  
NIKHIL KUMAR(19SCSE1010532)

# Table of Contents

	<b>Page No.</b>
<b>1. Abstract .....</b>	<b>4</b>
<b>2. Introduction .....</b>	<b>5</b>
<b>3. Feasibility Study.....</b>	<b>6</b>
<b>4. Objective of the Project.....</b>	<b>9</b>
<b>5. System Design.....</b>	<b>10</b>
<b>6. Problem Statement .....</b>	<b>12</b>
<b>7. Methodology .....</b>	<b>13</b>
<b>8. Implementation Details.....</b>	<b>16</b>
<b>9. Conclusion.....</b>	<b>30</b>
<b>10. References.....</b>	<b>31</b>

# 1. Abstract

Sentiment Analysis also known as Opinion Mining refers to the use of natural language processing, text analysis to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine.

In this project, we aim to perform Sentiment Analysis from Text Feedback. Data used in this project are online product reviews collected from “Restaurant Review”. We expect to do review-level categorization of review data with promising outcomes.

## 2. Introduction

Sentiment is an attitude, thought, or judgment prompted by feeling. Sentiment analysis, which is also known as opinion mining, studies people's sentiments towards certain entities. From a user's perspective, people are able to post their own content through various social media, such as forums, micro-blogs, or online social networking sites. From a researcher's perspective, many social media sites release their application programming interfaces (APIs), prompting data collection and analysis by researchers and developers. However, those types of online data have several flaws that potentially hinder the process of sentiment analysis. The first flaw is that since people can freely post their own content, the quality of their opinions cannot be guaranteed. The second flaw is that ground truth of such online data is not always available. A ground truth is more like a tag of a certain opinion, indicating whether the opinion is positive, negative, or neutral.

“It is a quite boring movie but the scenes were good enough.”

The given line is a movie review that states that “it” (the movie) is quite boring but the scenes were good. Understanding such sentiments require multiple tasks.

Hence, SENTIMENTAL ANALYSIS is a kind of text classification based on Sentimental Orientation (SO) of opinion they contain.

Sentiment analysis of product reviews has recently become very popular in text mining and computational linguistics research.

- Firstly, evaluative terms expressing opinions must be extracted from the review.
- Secondly, the SO, or the polarity, of the opinions must be determined.
- Thirdly, the opinion strength, or the intensity, of an opinion should also be determined.
- Finally, the review is classified with respect to sentiment classes, such as Positive and Negative, based on the SO of the opinions it contains.

## **3. Feasibility Study**

### **3.1 Feasibility study**

Before developing this project, we first analyse existed system of study. In existed system all work is performed using papers. As we know, now a day computer is used in every field. We can remove the paper work by using automatic system. We see it first that if it is feasible or not whether technically, economically, operationally. We test that whether it properly works or not. Its technical requirements are feasible or not. We analysed the system properly and then start designing it. After designing, we implement this project that whether this project works properly or not. After implementing the project, we check that whether there is any problem for the user while using this project.

### **3.2 System feasibility**

Prior to stating whether the system we have to develop is feasible or not we believe that we should emphasize on what is implied by the word “Feasibility”. Feasibility is the measure of how beneficial practical the development of the system will be to the organization. It is a preliminary survey for the systems investigation.

### **3.3 Types**

There are various measures of feasibility that helps to decide whether a particular project is feasible or not. These measures include –

1. Operational Feasibility
2. Technical Feasibility
3. Economic Feasibility

#### **3.3.1 Operational feasibility**

A proposed system is beneficial only if it can be turned into an information system that will meet the operational requirements of an organization. A system often fails if it does not fit within existing operations and if users resist the change. Important issues a systems developer must look into are:

Will the new system be used if implemented in an organization?

Are there any major barriers to implementation or is proposed system accepted without destructive resistance?

The whole purpose of computerizing the Complaint Management is to handle the work much more accurately and efficiently with less time consumption.

There will be additional work to be completed, because now the cellular company will have to maintain database of both their employees as well as their Customers. Compared to the semi-computerized system the chances of avoiding errors in a computerized system is much higher because the user need not stress himself unnecessarily resulting in recklessness. Unlike the semi-computerized system there would be backup data for all the information concerning the daily transactions occurred within the organization. Another important fact to be regarded is the security control, which is handled by the system. Since data regarding each Customer and the Organization is confidential, security is a key issue. Information falling into the wrong hands could jeopardize the entire organization. Unlike in semi-computerized systems. The proposed system offers adequate control to protect the organization against fraud and embezzlement and guarantees the accuracy and Security of data and information. This is handled by the system providing individuals with separate login names and passwords.

### **3.3.2 Technical feasibility**

Based on the outline design of the system requirements in terms of inputs, output, Procedures, the technical issues raised during technical feasibility include:

Does the necessary technology exist to do what is proposed?

Does the proposed equipment have the technical capacity to hold the data required to use in the new system?

Adequate responses provided by the proposed system? Is the system flexible enough to facilitate expansion?

Is there any technical guarantee of accuracy, reliability, ease of access and data security?

The system developer's task is to view needed capabilities in light of currently available technology. Our site works hand in hand with high technology. A database has to be maintained in order to update and backup data whenever required. To create databases, we use MySQL server. After taking the above.

### **3.3.3 Economic feasibility**

In making recommendations a study of the economics of the proposed system should be made. Even though finding out the costs of the proposed project is difficult we assume and estimate the costs and benefits as follows.

According to the computerized system we propose, the costs can be broken down in two categories.

- Costs associated with the development of the system.
- Costs associated with operating the system



## 4. Objective of the Project

- Scrapping product reviews on various websites featuring various products specifically amazon.com.
- Analyze and categorize review data.
- Analyze sentiment on dataset from document level (review level).
- Categorization or classification of opinion sentiment into-
  - Positive
  - Negative

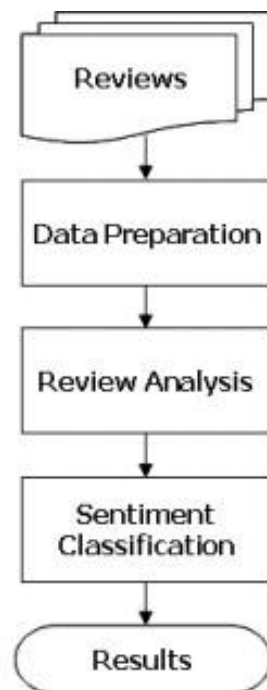


Figure 1: A typical sentiment analysis model.

## 5. System Design

### Hardware Requirements:

- Core i3 processor
- At least 4 GB RAM
- At least 60 GB of Usable Hard Disk Space

### Software Requirements:

- Python 3.x
- Anaconda Distribution
- Window Operating System.

### Data Information:

- The Amazon reviews dataset consists of reviews from amazon. The data span a period of 18 years, including ~35 million reviews up to March 2013. Reviews include product and user information, ratings, and a plaintext review. For more information, please refer to the following paper: J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. RecSys, 2013.
- The Amazon reviews full score dataset is constructed by Xiang Zhang (xiang.zhang@nyu.edu) from the above dataset. It is used as a text classification benchmark in the following paper: Xiang Zhang, Junbo Zhao, Yann LeCun. Character-level Convolutional Networks for Text Classification. Advances in Neural Information Processing Systems 28 (NIPS 2015).
- The Amazon reviews full score dataset is constructed by randomly taking 200,000 samples for each review score from 1 to 5. In total there are 1,000,000 samples.

Star Level	General Meaning
★	I hate it.
★★	I don't like it.
★★★	It's okay.
★★★★	I like it.
★★★★★	I love it.

Books	<a href="#">reviews</a> (22,507,155 reviews)	<a href="#">metadata</a> (2,370,585 products)	<a href="#">image features</a>
Electronics	<a href="#">reviews</a> (7,824,482 reviews)	<a href="#">metadata</a> (498,196 products)	<a href="#">image features</a>
Movies and TV	<a href="#">reviews</a> (4,607,047 reviews)	<a href="#">metadata</a> (208,321 products)	<a href="#">image features</a>
CDs and Vinyl	<a href="#">reviews</a> (3,749,004 reviews)	<a href="#">metadata</a> (492,799 products)	<a href="#">image features</a>
Clothing, Shoes and Jewelry	<a href="#">reviews</a> (5,748,920 reviews)	<a href="#">metadata</a> (1,503,384 products)	<a href="#">image features</a>
Home and Kitchen	<a href="#">reviews</a> (4,253,926 reviews)	<a href="#">metadata</a> (436,988 products)	<a href="#">image features</a>
Kindle Store	<a href="#">reviews</a> (3,205,467 reviews)	<a href="#">metadata</a> (434,702 products)	<a href="#">image features</a>
Sports and Outdoors	<a href="#">reviews</a> (3,268,695 reviews)	<a href="#">metadata</a> (532,197 products)	<a href="#">image features</a>
Cell Phones and Accessories	<a href="#">reviews</a> (3,447,249 reviews)	<a href="#">metadata</a> (346,793 products)	<a href="#">image features</a>
Health and Personal Care	<a href="#">reviews</a> (2,982,326 reviews)	<a href="#">metadata</a> (263,032 products)	<a href="#">image features</a>
Toys and Games	<a href="#">reviews</a> (2,252,771 reviews)	<a href="#">metadata</a> (336,072 products)	<a href="#">image features</a>
Video Games	<a href="#">reviews</a> (1,324,753 reviews)	<a href="#">metadata</a> (50,953 products)	<a href="#">image features</a>
Tools and Home Improvement	<a href="#">reviews</a> (1,926,047 reviews)	<a href="#">metadata</a> (269,120 products)	<a href="#">image features</a>
Beauty	<a href="#">reviews</a> (2,023,070 reviews)	<a href="#">metadata</a> (259,204 products)	<a href="#">image features</a>
Apps for Android	<a href="#">reviews</a> (2,638,173 reviews)	<a href="#">metadata</a> (61,551 products)	<a href="#">image features</a>
Office Products	<a href="#">reviews</a> (1,243,186 reviews)	<a href="#">metadata</a> (134,838 products)	<a href="#">image features</a>
Pet Supplies	<a href="#">reviews</a> (1,235,316 reviews)	<a href="#">metadata</a> (110,707 products)	<a href="#">image features</a>
Automotive	<a href="#">reviews</a> (1,373,768 reviews)	<a href="#">metadata</a> (331,090 products)	<a href="#">image features</a>
Grocery and Gourmet Food	<a href="#">reviews</a> (1,297,156 reviews)	<a href="#">metadata</a> (171,760 products)	<a href="#">image features</a>
Patio, Lawn and Garden	<a href="#">reviews</a> (993,490 reviews)	<a href="#">metadata</a> (109,094 products)	<a href="#">image features</a>
Baby	<a href="#">reviews</a> (915,446 reviews)	<a href="#">metadata</a> (71,317 products)	<a href="#">image features</a>
Digital Music	<a href="#">reviews</a> (836,006 reviews)	<a href="#">metadata</a> (279,899 products)	<a href="#">image features</a>
Musical Instruments	<a href="#">reviews</a> (500,176 reviews)	<a href="#">metadata</a> (84,901 products)	<a href="#">image features</a>
Amazon Instant Video	<a href="#">reviews</a> (583,933 reviews)	<a href="#">metadata</a> (30,648 products)	<a href="#">image features</a>

## 6. Problem Statement

- Sentiment analysis is contextual mining of text which identifies and extracts subjective information in source material, and helping a business to understand the social sentiment of their brand, product and improve their product.
- Web Portals get a vast amount of feedback from the users. To go through all the feedback can be a tedious job. Develop a software to categorize opinions expressed in feedback forums. This can be utilized for a feedback management system. The software must provide the classification of individual comments/reviews to allow the owners to improve their service by taking them into consideration.
- In today's world where we are suffering from data overload companies might have vast amount of customer feedback collected. Yet for we humans, it's still impossible to analyze it manually without any sort of error or bias. Oftentimes, companies with the best intentions find themselves in an insights vacuum. You know you need insights to inform your decision making. And you know that you're lacking them. But you don't know how best to get them. Sentiment analysis provides answers into what the most important issues are. Because sentiment analysis can be automated, decisions can be made based on a significant amount of data rather than plain intuition that isn't always right.

## 7. Methodology

- Training our model with a restaurant review dataset
- Analyze and categorize review data.
- Analysis the sentiment on dataset.
- Classification of opinion sentiment into-
  - Positive
  - Negative

### DATA COLLECTION:

Data which means product reviews collected from amazon.com from May 1996 to July 2014. Each review includes the following information: 1) reviewer ID; 2) product ID; 3) rating; 4) time of the review; 5) helpfulness; 6) review text. Every rating is based on a 5-star scale, resulting all the ratings to be ranged from 1-star to 5-star with no existence of a half-star or a quarter-star.

### SENTIMENT SENTENCE EXTRACTION & POS TAGGING:

Tokenization of reviews after removal of STOP words which mean nothing related to sentiment is the basic requirement for POS tagging. After proper removal of STOP words like “am, is, are, the, but” and so on the remaining sentences are converted in tokens. These tokens take part in POS tagging

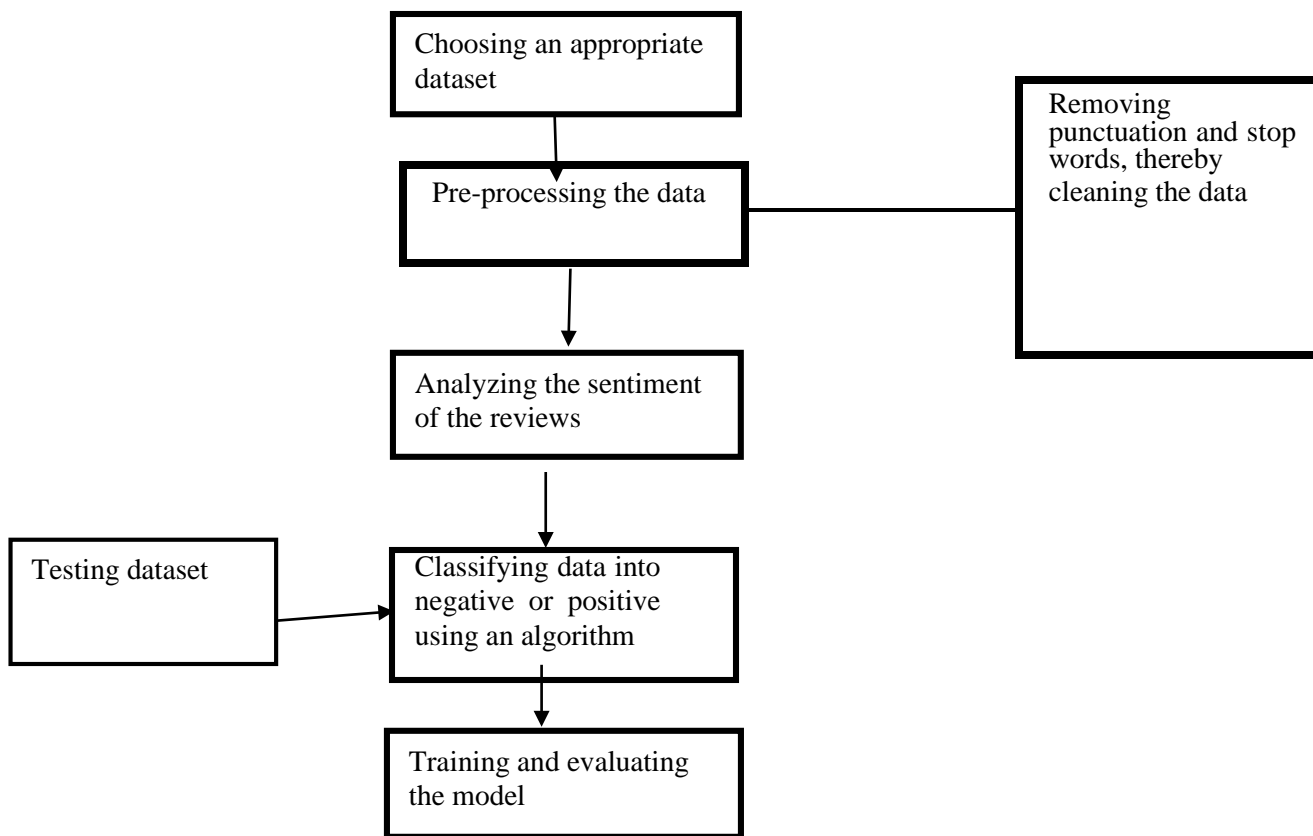
In natural language processing, part-of-speech (POS) taggers have been developed to classify words based on their parts of speech. For sentiment analysis, a POS tagger is very useful because of the following two reasons: 1) Words like nouns and pronouns usually do not contain any sentiment. It is able to filter out such words with the help of a POS tagger; 2) A POS tagger can also be used to distinguish words that can be used in different parts of speech.

### NEGATIVE PHRASE IDENTIFICATION:

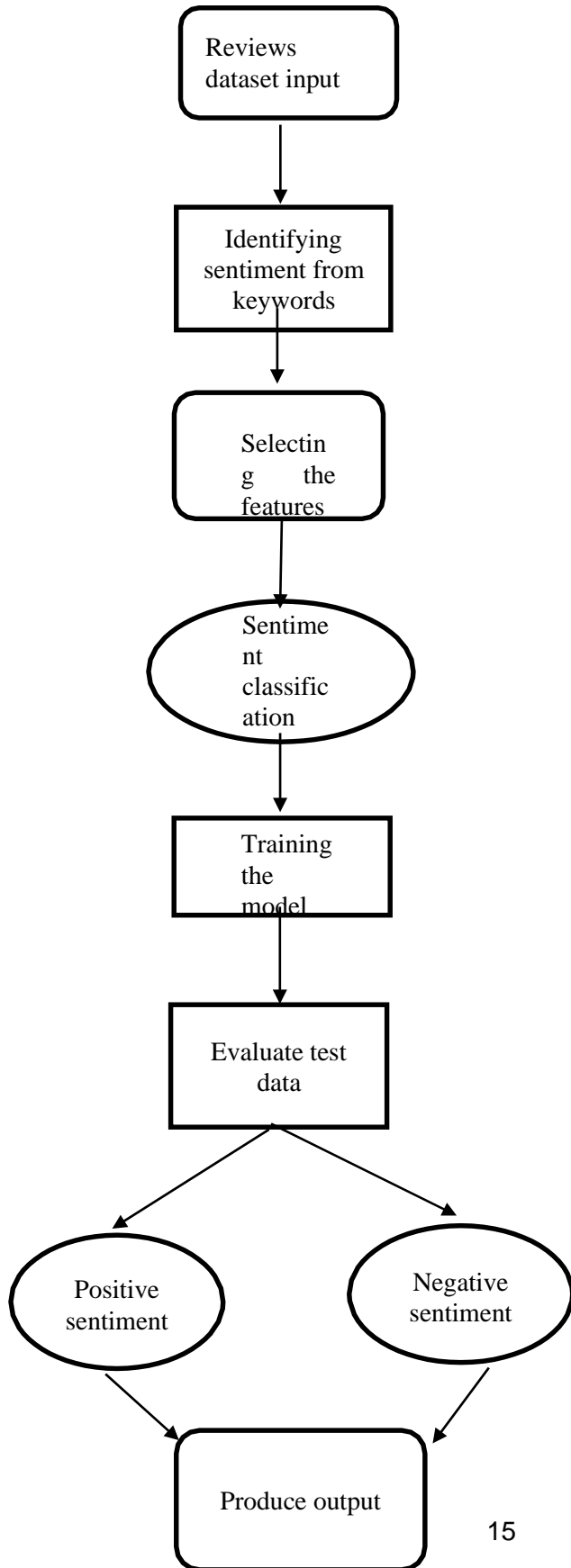
Words such as adjectives and verbs are able to convey opposite sentiment with the help of negative prefixes. For instance, consider the following sentence that was found in an electronic device’s review: “The built in speaker also has its uses but so far nothing

revolutionary." The word, "revolutionary" is a positive word according to the list in. However, the phrase "nothing revolutionary" gives more or less negative feelings. Therefore, it is crucial to identify such phrases. In this work, there are two types of phrases have been identified, namely negation-of-adjective (NOA) and negation-of-verb (NOV).

**Architecture Diagram:**



**Flow Diagram:**



## 8. Implementation Details

The training of dataset consists of the following steps:

- ✚ **Unpacking of data:** The huge dataset of reviews obtained from amazon.com comes in a .json file format. A small python code has been implemented in order to read the dataset from those files and dump them in to a pickle file for easier and fastaccess and object serialization.

```
30 with open(data_file, 'r') as file_handler:
31     for review in file_handler.readlines():
32         df[i] = ast.literal_eval(review)
33         i += 1
34
35 reviews_df = pd.DataFrame.from_dict(df, orient = 'index')
36 reviews_df.to_pickle('reviews_digital_music.pickle')
37
```

Hence initial fetching of data is done in this section using Python File Handlers.

- ✚ Preparing Data for Sentiment Analysis:

**i)** The pickle file is hence loaded in this step and the data besides the one used for sentiment analysis is removed. As shown in our sample dataset in Page 11, there are a lot of columns in the data out of which only rating and text review is what we require. So, the column, “reviewSummary” is dropped from the data file.

**ii)** After that, the review ratings which are 3 out of 5 are removed as they signify neutral review, and all we are concerned of is positive and negative reviews.

**iii)** *The entire task of preprocessing the review data is handled by this*

```
40
47 reviews_df.drop(columns = ['reviewSummary'], inplace = True)
48 reviews_df['reviewRating'] = reviews_df.reviewRating.astype('int')
49
50 reviews_df = reviews_df[reviews_df.reviewRating != 3] # Ignoring 3-star reviews -> neutral
51 reviews_df = reviews_df.assign(sentiment = np.where(reviews_df['reviewRating'] >= 4, 1, 0)) # 1 -> Positive, 0 -> Negati
52
```



utility class- “NltkPreprocessor”.

```
16
17 class NltkPreprocessor:
18
19     def __init__(self, stopwords = None, punct = None, lower = True, strip = True):
20         self.lower = lower
21         self.strip = strip
22         self.stopwords = stopwords or set(sw.words('english'))
23         self.punct = punct or set(string.punctuation)
24         self.lemmatizer = WordNetLemmatizer()
25
26     def tokenize(self, document):
27         tokenized_doc = []
28
29         for sent in sent_tokenize(document):
30             for token, tag in pos_tag(wordpunct_tokenize(sent)):
31                 token = token.lower() if self.lower else token
32                 token = token.strip() if self.strip else token
33                 token = token.strip('_0123456789') if self.strip else token
34                 # token = re.sub(r'\d+', '', token)
35
36                 if token in self.stopwords:
37                     continue
38
39                 if all(char in self.punct for char in token):
40                     continue
41
42                 lemma = self.lemmatize(token, tag)
43                 tokenized_doc.append(lemma)
44
45         return tokenized_doc
46
47     def lemmatize(self, token, tag):
48         tag = {
49             'N': wn.NOUN,
50             'V': wn.VERB,
51             'R': wn.ADV,
52             'J': wn.ADJ
53         }.get(tag[0], wn.NOUN)
54
55         return self.lemmatizer.lemmatize(token, tag)
56
```

iv) The time required to prepare the following data is hence displayed.


```
administrator@administrator-OptiPlex-3040:~/Desktop/sentiment_analysis$
Preprocessing data...
Preprocessing data completed!
Preprocessing time: 0.163 s
```

The time taken to preprocess the data is calculated and displayed

✚ **Preprocessing Data:** This is a vital part of training the dataset. Here Words present in the file are accessed both as a solo word and also as pair of words. Because, for example the word “bad” means negative but when someone writes “not bad” it refers to as positive. In such cases considering single word for training data will work otherwise. So words in pairs are checked to find the occurrence to modifiers before

any adjective which if present which might provide a different meaning to the outlook.

```
69 X = reviews_df_preprocessed.iloc[:, -1].values
70 y = reviews_df_preprocessed.iloc[:, -2].values
71
72 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
73
```

 **Training Data/ Evaluation:**The main chunk of code that does the whole evaluation of sentimental analysis based on the preprocessed data is a part of this. The following are the steps followed:

```
103 pipeline = Pipeline([
104     ('vect', TfidfVectorizer(ngram_range = (1,2), stop_words = 'english', sublinear_tf = True)),
105     ('chi', SelectKBest(score_func = chi2, k = 50000)),
106     ('clf', LinearSVC(C = 1.0, penalty = 'l1', max_iter = 3000, dual = False, class_weight='balanced'))
107 ])
108
109 model = pipeline.fit(X_train, y_train)
110
```

- i) The Accuracy, Precision, Recall, and Evaluation time is calculated and displayed.
- ii) Prediction of test data is done and Confusion Matrix of prediction is displayed.
- iii) Total positive and negative reviews are counted.
- iv) A review like sentence is taken as input on the console and if positive the console gives 1 as output and 0 for negative input.

## **Code Description :**

### Loading the dataset:

```
import json
import pickle
import numpy as np
from matplotlib import pyplot as plt
from textblob import TextBlob

# fileHandler = open('datasets/reviews_digital_music.json', 'r')
# reviewDatas = fileHandler.read().split('\n')
# reviewText = []
# reviewRating = []

# for review in reviewDatas:
#     if review == "":
#         continue
#     r = json.loads(review)
#     reviewText.append(r['reviewText'])
#     reviewRating.append(r['overall'])

# fileHandler.close()
# saveReviewText = open('review_text.pkl', 'wb')
# saveReviewRating = open('review_rating.pkl','wb')
# pickle.dump(reviewText, saveReviewText)
# pickle.dump(reviewRating, saveReviewRating)
reviewTextFile = open('review_text.pkl', 'rb')
```

```

reviewRatingFile = open('review_rating.pkl', 'rb')
reviewText = pickle.load(reviewTextFile)
reviewRating = pickle.load(reviewRatingFile)
# print(len(reviewText))
# print(reviewText[0])
# print(reviewRating[0])
# ratings = np.array(reviewRating)

plt.hist(ratings, bins=np.arange(ratings.min(), ratings.max()+2)-0.5, rwidth=0.7)
plt.xlabel('Rating', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.title('Histogram of Ratings', fontsize=18)
plt.show()

lang = { }
i = 0
for review in reviewText:
    tb = TextBlob(review)
    l = tb.detect_language()
    if l != 'en':
        lang.setdefault(l, [])
        lang[l].append(i)
        print(i, l)
    i += 1
print(lang)

```

### **Scrapping data:**

```

from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from bs4 import BeautifulSoup
import openpyxl

class Review():
    def __init__(self):

```

```

        self.rating=""
        self.info=""
        self.review=""
def scrape():
    options = Options()
    options.add_argument("--headless") # Runs Chrome in headless mode.
    options.add_argument('--no-sandbox') # # Bypass OS security model
    options.add_argument('start-maximized')
    options.add_argument('disable-infobars')
    options.add_argument("--disable-extensions")
    driver=webdriver.Chrome(executable_path=r'C:\chromedriver\chromedriver.exe')
    url='https://www.amazon.com/Moto-PLUS-5th-Generation-Exclusive/product-reviews/B0785NN142/ref=cm_cr_arp_d_paging_btm_2?ie=UTF8&reviewerType=all_reviews&pageNumber=5'
    driver.get(url)

    soup=BeautifulSoup(driver.page_source,'lxml')
    ul=soup.find_all('div',class_='a-section review')
    review_list=[]
    for d in ul:
        a=d.find('div',class_='a-row')
        sib=a.findNextSibling()
        b=d.find('div',class_='a-row a-spacing-medium review-data')
        "print sib.text"
        new_r=Review()
        new_r.rating=a.text
        new_r.info=sib.text
        new_r.review=b.text

        review_list.append(new_r)
    driver.quit()
    return review_list
def main():

```

```

m = scrape()
i=1
for r in m:

    book = openpyxl.load_workbook('Sample.xlsx')
    sheet = book.get_sheet_by_name('Sample Sheet')
    sheet.cell(row=i, column=1).value = r.rating
    sheet.cell(row=i, column=1).alignment = openpyxl.styles.Alignment(horizontal='center',
vertical='center', wrap_text=True)
    sheet.cell(row=i, column=3).value = r.info
    sheet.cell(row=i, column=3).alignment = openpyxl.styles.Alignment(horizontal='center',
vertical='center', wrap_text=True)
    sheet.cell(row=i, column=5).value = r.review.encode('utf-8')
    sheet.cell(row=i, column=5).alignment = openpyxl.styles.Alignment(horizontal='center',
vertical='center', wrap_text=True)
    book.save('Sample.xlsx')
    i=i+1
if __name__ == '__main__':
    main()

```

### Preprocessing Data:

```

import string
from nltk.corpus import stopwords as sw
from nltk.corpus import wordnet as wn
from nltk import wordpunct_tokenize
from nltk import sent_tokenize
from nltk import WordNetLemmatizer
from nltk import pos_tag
class NltkPreprocessor:
    def __init__(self, stopwords = None, punct = None, lower = True, strip = True):
        self.lower = lower
        self.strip = strip
        self.stopwords = stopwords or set(sw.words('english'))

```

```

self.punct = punct or set(string.punctuation)
self.lemmatizer = WordNetLemmatizer()

def tokenize(self, document):
    tokenized_doc = []

    for sent in sent_tokenize(document):
        for token, tag in pos_tag(wordpunct_tokenize(sent)):
            token = token.lower() if self.lower else token
            token = token.strip() if self.strip else token
            token = token.strip('_0123456789') if self.strip else token
            # token = re.sub(r'\d+', '', token)

            if token in self.stopwords:
                continue

            if all(char in self.punct for char in token):
                continue

            lemma = self.lemmatize(token, tag)
            tokenized_doc.append(lemma)

    return tokenized_doc

def lemmatize(self, token, tag):
    tag = {
        'N': wn.NOUN,
        'V': wn.VERB,
        'R': wn.ADV,
        'J': wn.ADJ
    }.get(tag[0], wn.NOUN)
    return self.lemmatizer.lemmatize(token, tag)

```

## Sentiment Analysis:

```
import ast
import numpy as np
import pandas as pd
import re
from nltk.corpus import stopwords
from nltk.stem import SnowballStemmer
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest, chi2, SelectPercentile, f_classif
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score,
confusion_matrix
from sklearn.svm import LinearSVC
# from textblob import TextBlob
from time import time

def getInitialData(data_file):
    print('Fetching initial data...')
    t = time()

    i = 0
    df = {}
    with open(data_file, 'r') as file_handler:
        for review in file_handler.readlines():
            df[i] = ast.literal_eval(review)
            i += 1

    reviews_df = pd.DataFrame.from_dict(df, orient = 'index')
    reviews_df.to_pickle('reviews_digital_music.pickle')
```



```

print('Fetching data completed!')
print('Fetching time: ', round(time()-t, 3), '\n')

# def filterLanguage(text):
#     text_blob = TextBlob(text)
#     return text_blob.detect_language()

def prepareData(reviews_df):
    print('Preparing data...')
    t = time()

    reviews_df.rename(columns = {"overall" : "reviewRating"}, inplace=True)
    reviews_df.drop(columns = ['reviewerID', 'asin', 'reviewerName', 'helpful', 'summary',
'unixReviewTime', 'reviewTime'], inplace = True)

    reviews_df = reviews_df[reviews_df.reviewRating != 3.0] # Ignoring 3-star reviews -> neutral
    reviews_df = reviews_df.assign(sentiment = np.where(reviews_df['reviewRating'] >= 4.0, 1, 0)) # 1
-> Positive, 0 -> Negative

    stemmer = SnowballStemmer('english')
    stop_words = stopwords.words('english')

    # print(len(reviews_df.reviewText))
    # filterLanguage = lambda text: TextBlob(text).detect_language()
    # reviews_df = reviews_df[reviews_df[reviewText].apply(filterLanguage) == 'en']
    # print(len(reviews_df.reviewText))

    reviews_df = reviews_df.assign(cleaned = reviews_df[reviewText].apply(lambda text: '
.join([stemmer.stem(w) for w in re.sub('[^a-z]+|(quot)+', ' ', text.lower()).split() if w not in stop_words]))
    reviews_df.to_pickle('reviews_digital_music_preprocessed.pickle')

```

```

print('Preparing data completed!')
print('Preparing time: ', round(time()-t, 3), 's\n')

def preprocessData(reviews_df_preprocessed):
    print('Preprocessing data...')
    t = time()

    X = reviews_df_preprocessed.iloc[:, -1].values
    y = reviews_df_preprocessed.iloc[:, -2].values

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)

    print('Preprocessing data completed!')
    print('Preprocessing time: ', round(time()-t, 3), 's\n')

    return X_train, X_test, y_train, y_test

def evaluate(y_test, prediction):
    print('Evaluating results...')
    t = time()

    print('Accuracy: {}'.format(accuracy_score(y_test, prediction)))
    print('Precision: {}'.format(precision_score(y_test, prediction)))
    print('Recall: {}'.format(recall_score(y_test, prediction)))
    print('f1: {}'.format(f1_score(y_test, prediction)))

    print('Results evaluated!')
    print('Evaluation time: ', round(time()-t, 3), 's\n')

# getInitialData('datasets/reviews_digital_music.json')
# reviews_df = pd.read_pickle('reviews_digital_music.pickle')

```

```

# prepareData(reviews_df)
reviews_df_preprocessed = pd.read_pickle('reviews_digital_music_preprocessed.pickle')
# print(reviews_df_preprocessed.isnull().values.sum()) # Check for any null values

X_train, X_test, y_train, y_test = preprocessData(reviews_df_preprocessed)

print('Training data...')
t = time()

pipeline = Pipeline([
    ('vect', TfidfVectorizer(ngram_range = (1,2), stop_words = 'english',
sublinear_tf = True)),
    ('chi', SelectKBest(score_func = chi2, k = 50000)),
    ('clf', LinearSVC(C = 1.0, penalty = 'l1', max_iter = 3000, dual = False,
class_weight = 'balanced'))
])

model = pipeline.fit(X_train, y_train)

print('Training data completed!')
print('Training time: ', round(time()-t, 3), 's\n')

print('Predicting Test data...')
t = time()

prediction = model.predict(X_test)

print('Prediction completed!')
print('Prediction time: ', round(time()-t, 3), 's\n')

evaluate(y_test, prediction)

print('Confusion matrix: {}'.format(confusion_matrix(y_test, prediction)))

```

```

print()
l = (y_test == 0).sum() + (y_test == 1).sum()
s = y_test.sum()
print('Total number of observations: ' + str(l))
print('Positives in observation: ' + str(s))
print('Negatives in observation: ' + str(l - s))
print('Majority class is: ' + str(s / l * 100) + '%')

```

Graph Plotting Code:

```

import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import MaxNLocator
from collections import namedtuple

n_groups = 5

score_MNB = (85.25, 85.31, 85.56, 84.95, 85.31)
score_LR = (88.12, 88.05, 87.54, 88.72, 88.05)
score_L SVC=(88.12, 88.11, 87.59, 88.80, 88.11)
score_RF=(82.43, 81.82, 79.74, 85.30, 81.83)

#n1=(score_MNB[0], score_LR[0], score_L SVC[0], score_RF[0])
#n2=(score_MNB[1], score_LR[1], score_L SVC[1], score_RF[1])
#n3=(score_MNB[2], score_LR[2], score_L SVC[2], score_RF[2])
#n4=(score_MNB[3], score_LR[3], score_L SVC[3], score_RF[3])
#n5=(score_MNB[4], score_LR[4], score_L SVC[4], score_RF[4])

fig, ax = plt.subplots()

index = np.arange(n_groups)

bar_width = 0.1

opacity = 0.7

error_config = {'ecolor': '0.3'}

rects1 = ax.bar(index,score_MNB, bar_width,
                 alpha=opacity, color='b',

```

```

        error_kw=error_config, label='Multinomial
        Naive Bayes')
z=index + bar_width
rects2 = ax.bar(z, score_LR, bar_width,
                alpha=opacity, color='r',
                error_kw=error_config,
                label='Logistic Regression')
z=z+ bar_width
rects3 = ax.bar(z, score_L SVC, bar_width,
                alpha=opacity, color='y',
                error_kw=error_config,
                label='Linear SVM')
z=z+ bar_width
rects4 = ax.bar(z, score_RF, bar_width,
                alpha=opacity, color='g',
                error_kw=error_config,
                label='Random Forest')
ax.set_xlabel('Score Parameters')
ax.set_ylabel('Scores (in %)')
ax.set_title('Scores of Classifiers')
ax.set_xticks(index + bar_width / 2)
ax.set_xticklabels(('F1', 'Accuracy', 'Precision', 'Recall', 'ROC AUC'))
ax.legend(bbox_to_anchor=(1, 1.02), loc=5, borderaxespad=0) fig.tight_layout()
plt.show()

```

## **9. CONCLUSION**

Sentiment analysis deals with the classification of texts based on the sentiments they contain. This article focuses on a typical sentiment analysis model consisting of three core steps, namely data preparation, review analysis and sentiment classification, and describes representative techniques involved in those steps.

Sentiment analysis is an emerging research area in text mining and computational linguistics, and has attracted considerable research attention in the past few years. Future research shall explore sophisticated methods for opinion and product feature extraction, as well as new classification models that can address the ordered labels property in rating inference. Applications that utilize results from sentiment analysis is also expected to emerge in the near future.

## 10. References

1. <https://towardsdatascience.com/social-media-sentiment-analysis-part-ii-bcacca5aaa39>
2. <https://www.datacamp.com/community/tutorials/simplifying-sentiment-analysis-python>
3. <https://towardsdatascience.com/how-to-use-streamlit-to-deploy-your-ml-models-wrapped-in-beautiful-web-apps-66e07c3dd525>
4. <https://www.freecodecamp.org/news/how-to-deploy-a-nodejs-app-to-heroku-from-github-without-installing-heroku-on-your-machine-433bec770efe/>
5. Dave, D., Lawrence, A., and Pennock, D. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. Proceedings of International World Wide Web Conference (WWW'03), 2010.
6. Callen Rain, "Sentiment Analysis in Amazon Reviews Using Probabilistic Machine Learning" Swarthmore College, Department of Computer Science.