MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY

BHOPAL

Department of Computer Science & Engineering

MINOR PROJECT ON

FEATURE EXTRACTION FROM TEXT USING NATURAL LANGUAGE PROCESSING

Submitted in Partial Fulfilment for the degree of Bachelor of Technology

Submitted by:

RISHI ILLURI	161112235
R.CHAITANYA	161112272
B.SIVA RAMA KRISHNA	161112078
P.AKHIL REDDY	161112275

Under the guidance of:

Dr. Sri Khetwat Saritha

MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY BHOPAL

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that Rishi Illuri , Rangu Chaitanya, Bobbili Siva Rama Krishna, Akhil Reddy Poreddy students of B.Tech 3rd Year 6th Semester (Computer Science & Engineering), have successfully completed their project entitled **"Feature Extraction From Text using Natural Language Processing"** in partial fulfilment of their minor project in Computer Science & Engineering.

Dr. Sri Khetwat Saritha

(Project Guide)

MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY BHOPAL

Department of Computer Science & Engineering

DECLARATION

We, hereby, declare that the following report which is being presented in the **"Feature Extraction From Text using Natural Language Processing"** is the partial fulfillment of the requirements of the third year (sixth semester) Minor project in the field of Computer Science and Engineering. It is an authentic documentation of our own original work carried out under the able guidance of Dr. Sri Khetwat Saritha. The work has been carried out entirely at Maulana Azad National Institute of Technology, Bhopal. The following project and its report, in part or whole, has not been presented or submitted by us for any purpose in any other institute or organization.

We, hereby, declare that the facts mentioned above are true to the best of our knowledge. In case of any unlikely discrepancy that may possibly occur, we will be the ones to take responsibility.

RISHI ILLURI	161112235
R.CHAITANYA	161112272
B.SIVA RAMA KRISHNA	161112078
P.AKHIL REDDY	161112275

ACKNOWLEDGEMENT

With due respect, we express our deep sense of gratitude to our respected Mentor. It is imperative for us to mention the fact that this minor project could not have been accomplished without the periodic suggestions and advice of our project guide Prof S. K Saritha.

We are also grateful to our respected director Dr. Narendra Singh Raghuvanshi for permitting us to utilize all the necessary facilities of the college. Needless to mention is the additional help and support extended by our respected HOD, Dr. Meenu Chawla, in allowing us to use the departmental laboratories and other services.

We are also thankful to all the other faculty, staff members and laboratory attendants of our department for their kind cooperation and help.

Last but certainly not the least, we would like to express our deep appreciation towards our family members and batch mates for providing the much-needed support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Online shopping is more and more common nowadays. The growth in its popularity has led to increase in customer reviews that a product receives. A customer who has to choose the right product among the huge varieties of products, depends heavily on the product reviews to make a purchase decision. With great volume of product reviews, it become difficult for customers to go through all reviews to make an informed product choice. Nowadays customers look for features that can serve them specifically. But from the thousands of reviews, it is practically impossible for customers to identify the reviews which speak about the specific product feature. But it became more difficult from choosing thousands of reviews in order to simplify this feature level extraction has being developed, from the past few years it was being developed .

In this project, there are two main algorithms HAC (High Adjective Count) and MOS (Majority Opinion Score). HAC is for finding adjective_list and adjective&features_list. MOS is to find the adjective defining particular feature. Polarity will be taken from predefined libraries The score will be added to it and it will be represented in bar_graphs.

INTRODUCTION

The opinions of others have a significant influence in our daily decision-making process. These decisions range from buying a product such as a smartphone to making investments to choosing a school-all decisions that affect various aspects of our daily life. Before the Internet, people would seek opinions on products and services from sources such as friends, relatives, or consumer reports [10].

However, in the Internet era, it is much easier to collect diverse opinions from different people around the world. People look to review sites (e.g., CNET, Epinions.com), e-commerce sites (e.g., Amazon, eBay), online opinion sites (e.g., TripAdvisor, Rotten Tomatoes, Yelp) and social media (e.g., Facebook, Twitter) to get feedback on how a particular product or service may be perceived in the market.

Similarly, organizations use surveys, opinion polls, and social media as a mechanism to obtain feedback on their products and services. sentiment analysis or opinion mining is the computational study of opinions, sentiments, and emotions expressed in text. The use of sentiment analysis is becoming more widely leveraged because the information it yields can result in the monetization of products and services.

For example, by obtaining consumer feedback on a marketing campaign, an organization can measure the campaign's success or learn how to adjust it for greater success. Product feedback is also helpful in building better products, which can have a direct impact on revenue, as well as comparing competitor offerings.

This will describe the various types of sentiment classification, explore how to convert unstructured text into structured opinions, and address the current challenges in the field.

2.1 Sentiment Classification Levels

Sentiment analysis can occur at different levels:

1. Document level,
2. Sentence level
3. Aspect/feature level.

2.2 Document Level Classification

In this process, sentiment is extracted from the entire review, and a whole opinion is classified based on the overall sentiment of the opinion holder. The goal is to classify a review as positive, negative, or neutral.

Example

> "I bought an iPhone a few days ago.
> It is such a nice phone, although a little large.
> The touch screen is cool.
> The voice quality is clear too.
> I simply love it!"

Is the review classification positive or negative?

According to person perspective the classification is positive.

Document level classification works best when the document is written by a single person and expresses an opinion/sentiment on a single entity.

2.3 Sentence Level Classification

This process usually involves two steps [10]:

• Subjectivity classification of a sentence into one of two classes: objective and subjective
• Sentiment classification of subjective sentences into two classes: positive and negative

An objective sentence presents some factual information, while a subjective sentence expresses personal feelings, views, emotions, or beliefs. Subjective sentence identification can be achieved through different methods such as Naïve Bayesian classification. However, just knowing that sentences have a positive or negative opinion is not sufficient. This is an intermediate step that helps filter out sentences with no opinions and helps determine to an extent if sentiments about entities and their aspects are positive or negative. A subjective sentence may contain multiple opinions and subjective and factual clauses.

Example

"iPhone sales are doing well in this bad economy."

Sentiment classification at both the document and sentence levels are useful, but they do not find what people like or dislike, nor do they identify opinion targets.

2.4 Aspect/Feature Level Classification

In this process the goal is to identify and extract object features that have been commented on by the opinion holder and determine whether the opinion is positive, negative, or neutral.

Feature synonyms are grouped, and a feature-based summary of multiple reviews is produced.

LITERATURE REVIEW

Many interesting works exist that focus on extracting the opinions from the customer reviews. Some works focus on performing opinion mining to identify the semantic orientation of a review overall, whereas others focus on identifying and extracting the opinion words that will determine the semantic orientation. This line of work further divides into those who focus on the 3 opinion word identification and semantic orientation, and those who also employ features as an additional tool in representing the semantic orientation of a review. Our work is mostly related to the latter category, but we do provide an overview of related work in the other research areas as well.

In [3] the authors calculate the semantic orientation of words based on their semantic association with pre-determined positive and negative words.
The work presented in [4] proposes an unsupervised method to extract syntactic structures that specify the orientation of clauses for domain oriented semantic analysis.
In [7] the opinion words are classified individually and then the polarity of the opinion sentence is calculated by combining the individual opinion word polarity
while in [9] the sentiment of each sentence is analyzed by identifying the sentiment expressions and subject terms. Sometimes the opinions regarding the products may not be explicitly mentioned on the customer review sites but they exist in web blogs. Techniques to extract opinions contained in the blogs are proposed in [10].

In [5] the authors propose methods to determine the term subjectivity and term orientation using semi-supervised learning process while in [6] the orientation of the subjective terms is determined by utilizing the term definitions contained in the dictionaries. Most of the mentioned approaches, however, only lay stress on opinion words and do not consider the features. Moreover, they do not incorporate the proposed methodology in a broader opinion mining framework. A few works exist that perform sentence-level sentiment analysis (i.e. sentiment analysis that is using words but is not extracting representative features) .

PROPOSED ARCHITECTURE & METHODOLOGY

4.1 Feature Based Sentiment Analysis

Feature based sentiment analysis is based on identifying aspects of given target entities and estimating the sentiment polarity for each mentioned aspect. This can be decomposed into two tasks: aspect extraction and aspect sentiment classification.

Aspect extraction pertains to recognizing aspects of the entity, and more generally can be seen as an information extraction task. Aspect sentiment classification determines whether the opinions on different aspects are positive, negative or neutral.

Feature Extraction

To identify all the aspect terms present in a sentence, all highly frequent phrases across reviews (e.g. food) should be found and filtered by rules like "occurs right after sentiment word" (e.g. great food). Then a set of phrases that occur frequently can be built. Another approach is to determine all the aspects in advance and find them in the reviews. For a restaurant, the aspects could be: food, service, value, décor.

Example

"The food was great, but the service was slow."

Aspects: food, service Sentiments:

4.1.1 Sentiment Classification

Words express various kinds of sentiments that may be positive, negative, strong, or weak. To perform sentiment analysis, it is important to understand the polarity of words and classify sentiments into categories such as positive, negative, or neutral. This task can be accomplished through the use of sentiment lexicons.There are different types of sentiment lexicons available that have words classified as having positive or negative

sentiments.

1. Data pre processing
2. Extracting Potential Features
3. Finding polarity of Each Adjective
4. Finding Score For each Potential Feature
5. Representing each Feature vs Score (Bar Graph)

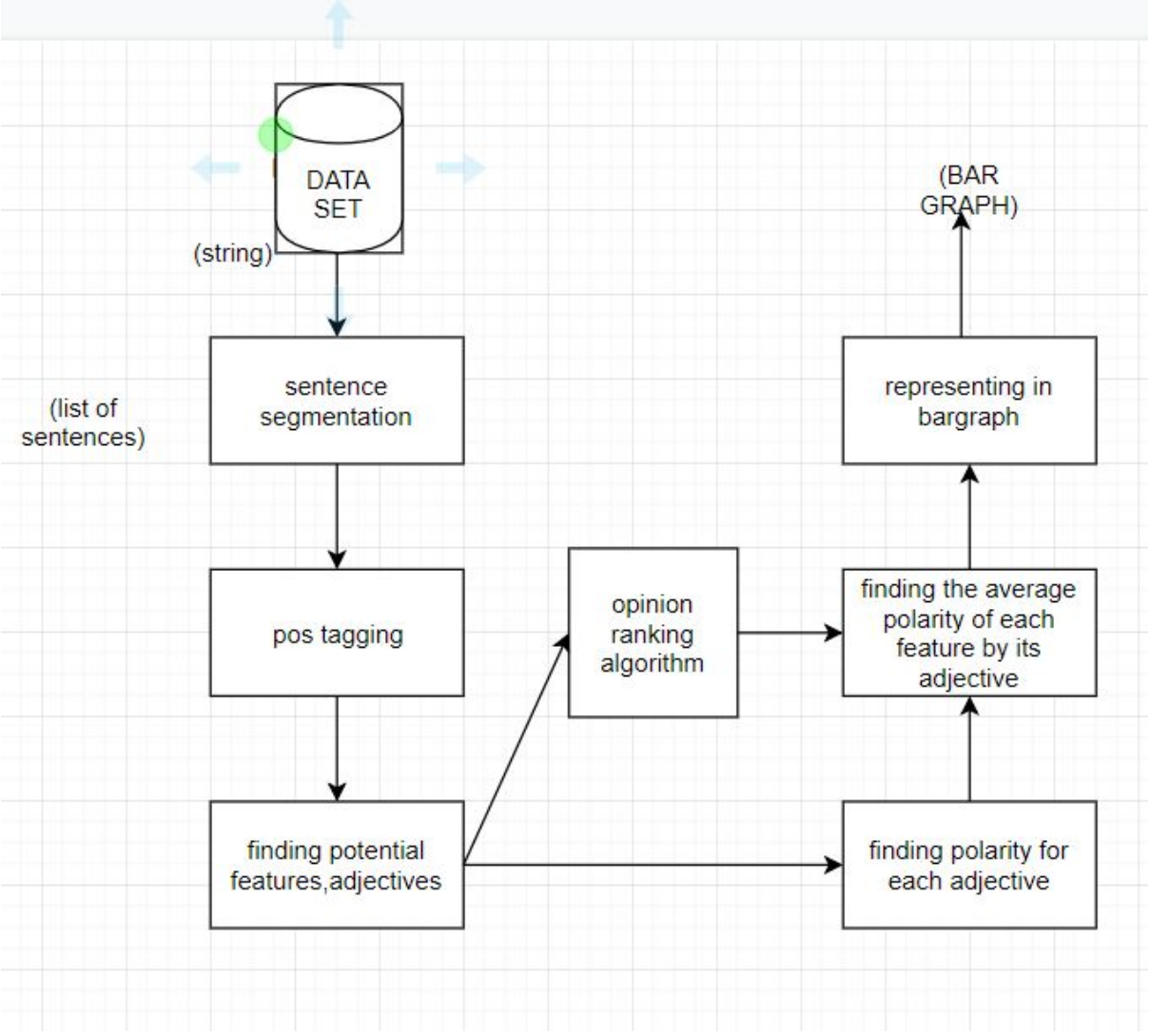

figure: 4.1 System Architecture

4.2.1 Data PreProcessing:

Each product review is pre-processed for sentence splitting, tokenization and Part-of-Speech (PoS) tagging. Stop Words, such as “a, the, it, their…..” from each review are removed. Most of the product features are nouns and most of the words used to determine the polarity of these features are adjectives found in the vicinity of the opinion feature. Hence, the information extracted from PoS tagging is of utmost importance as it helps us to focus more on the potential feature set and to exclude unnecessary information.

DataSet Format:

Example review of a mobile:

[+][t]excellent phone , excellent service .
##i am a business user who heavily depend on mobile service .
phone[+3], work[+2]##there is much which has been said in other reviews about the features of this phone , it is a great phone , mine worked without any problems right out of the box .

##just double check with customer service to ensure the number provided by amazon is for the city / exchange you wanted .

signal quality[+3]##i have owned motorola , panasonic and nokia phones over the last 8 years and generally prefer nokia , this phone combines many of the best nokia features , the only feature missing for me is the voice recognition .

speaker phone[+2],radio[+2],infrared[+2]##my favorite features, although there are many, are the speaker phone , the radio and the infrared .

speaker phone[+2]##the speaker phone is very functional and i use it in the car , very audible even with freeway noise .

infrared[+2]##the infrared is a blessing if you have a previous nokia and want to transfer your old phone book to this phone , saved me hours of re-entering my numbers .

[+] represents a positive review

[-] represents a negative review

Data From the data set is preprocessed as to set the data in the format which is acceptable to the data processing algorithm

For example:

Tag [t] is inserted at the beginning of the review to state that starting of the new review .

Review file of specific product is now divided into text containing each review separately.

Data After Sentence Segmentation:

```
[t]good phone , so-so service .
##the day finally arrived when i was sure i 'd leave sprint .
##after years with that carrier 's expensive plans and horrible customer service portability seemed heaven-sent .
##i 'd always eyed the nokia phones and had heard decent things about t-mobile so i gave it a whirl .
##here 's the brief synopsis the phone is tiny cute feels kind of plastic-like as if it might break but seems pretty sturdy .
##it has lots of little cute features my favorite being the games and the pim personal information manager i.e organizer and the radio .
##i spent hours setting up the stations accepts about 13-14 i believe though the reception is unpredictable .
##also you need to have the headset plugged in all the time to have the radiowork and that can get tedious .
##the headset that comes with the phone has good sound volume but it hurts the ears like you can not imagine .
##the phone comes with okay ringtones some decent backgrounds / screensavers but the phone has very little memory mine had 230kb as it ar
##the colors on the screen are not as crisp as i 'd have liked them to be .
##however it serves its purpose .
```

Snapshot 4.2 Example of sentence segmentation

Snapshot 4.2 shows the text from dataset after sentence segmentation

POSTags:

```
[]
[('ve', 'NN'), ('single', 'JJ'), ('bar', 'NN')]
[('battery', 'NN'), ('phone', 'NN'), ('small', 'JJ'), ('light', 'NN')]
[('phone', 'NN'), ('many', 'JJ'), ('name', 'NN'), ('beneficial', 'JJ')]
[('phone', 'NN'), ('minor', 'JJ'), ('high', 'JJ'), ('spend', 'NN'), ('internet', 'NN'), ('phone', 'NN')]
[('overall', 'JJ'), ('phone', 'NN')]
[('perfect', 'JJ'), ('phone', 'NN'), ('small', 'JJ'), ('appealing', 'NN'), ('package', 'NN')]
[]
```

Snapshot 4.3 POS tags

Snapshot 4.2 shows word and its POS tag

4.2.2 Extracting Potential Features

HAC Algorithm

- Instead of using frequency of keywords, the algorithm starts identifying adjectives and nouns in every review.
- For every review, if each adjective is associated with a noun to which it is closest, this adjective is more likely to describe the noun.
- If an adjective is found to be describing a noun, then the noun_score is incremented by 1. After processing all the reviews, we will have a score associated with each noun, which we call them as opinion scores.
- So nouns with high score have more adjectives to describe them. Then we can have a threshold and nouns having score more than threshold are considered as potential features.
- The adjectives are even extracted from this step .

Threshold is set to 3 and extracted features occurring more than 3 times

Procedure:

Features Using HAC(High Adjective Count) Algorithm are extracted

Preprocessed data is sent to HAC algorithm.

The HAC algorithm extracts the good potential features only when the reviews in the dataset follows the assumption that:

- Users comment on the features they care about using adjectives.
- The more often a noun is in the vicinity of an adjective, the more likely it is to be a representative feature of the item reviewed

For example

- If the review is

  "This camera has very good image quality"

  then the adjective 'good' is more likely to describe the noun "image quality" than the noun "camera".

```
[('lot', 44), ('cool', 33), ('free', 33)]
OrderedDict([('business', 0), ('mine', 0), ('number', 0), ('exchange', 0), ('week', 0),
('motorola', 0), ('recognition', 0), ('radio', 0), ('car', 0), ('book', 0), ('winner', 0),
('read', 0), ('box', 1), ('city', 1), ('look', 1), ('convert', 1), ('road', 1), ('trip',
1), ('reception', 1), ('i75', 1), ('route', 1), ('voice', 1), ('use', 1), ('noise', 1),
('combination', 1), ('price', 1), ('anything', 1), ('crystal', 1), ('service', 2),
('quality', 2), ('area', 2), ('cincinnati', 2), ('work', 2), ('nokia', 2), ('speaker', 2),
('transfer', 2), ('lot', 2), ('customer', 3), ('phone', 5)])
[]
########################
```

Snapshot 4.4 Each Potential Feature and its Count

Snapshot 4.4 shows each features and the frequency it appeared in review dataset

Potential features are words which potentially describes about features of a specific product. For example camera,battery,screen etc are the potential features of mobile.

```
{'mobile': 'text', 'reviews': 'phone', 'great': 'phone', 'right': 'realation', 'amazon':
'report', 'several': 'work', 'torture': 'customer', 'att': 'customer', 'drop': 'area',
'august': 'convert', 'signal': 'radio', 'detroit': 'area', 'suburbs': 'area', 'recent':
'road', 'northern': 'trip', 'kentucky': 'cincinnati', 'perfect': 'size', 'superior':
'phone', 's': 'radio', 'long': 'phone', 'panasonic': 'nokia', 'last': 'sony', 'nokia':
'phone', 'many': 'input', 'favorite': 'function', 'functional': 'device', 'audible':
'noise', 'infrared': 'port', 'previous': 'model', 'old': 'motorola', 'tmobile': 'phone',
'sure': 'd', 'expensive': 'cable', 'horrible': 'battery', 'decent': 'size', 'brief':
'synopsis', 'tiny': 'phone', 'pretty': 'sturdy', 'cute': 'information', 'personal':
'touch', 'unpredictable': 'reception', 'good': 'service', 'okay': 'phone', 'little':
'item', 'crisp': 'd', 'dangerous': 'night', 'small': 'size', 'uncreative': 'menu', 'full':
'bar', 'pick': 'scroll', 're': 'scroll', 'minor': 'phone', 'excellent': 'quality',
'quiet': 'line', 'public': 'transport', 'amazing': 'battery', 'assumed': 'service',
'exceptional': 'service', 'horrendous': 'los', 'cingular': 's', 'cheap': 'phone',
'martian': 'time', 'd': 'capability', 'wap': 'capability', 'fairness': 'customer', 'nice':
'phone', 'hard': 'press', 'expect': 've', 'answered': 'interaction', 'megastatic':
'point', 'dial': 'd', 'clear': 'sprint', 'worth': 'penny', 'bad': 'minus', 'nt': 'phone',
'able': 'carry', 'complete': 'working', 'fellow': 'effort', 'revive': 'phone', 'apparent':
```

Snapshot 4.5 Feature vs Adjective Dictionary

Snapchat 4.4 Shows dictionary of every feature and adjective defining it.

4.2.3 Finding polarity of Each Adjective

Polarity

Polarity is a positive or negative number assigned to every word. Positive polarity of a word shows that the word is positive similarly negative polarity shows that the word is negative.

Each adjective from all the reviews are stored in a dictionary and each adjectives are assigned with a polarity ranging from -4 to 4.

Create dictionary of adjectList

for i in adjectList:

blob= TextBlob(i)

If polarity not equal to zero:

adjScores[i]=polarity

Assign polarity from -4 to 4

Snapshot 4.6 Each Adjective and its Scores [-4,4]

Snapchat 4.3.1 shows polarity of each adjective that appears in review

4.2.4 Finding Score For each Potential Feature

After extracting potential features, these features and the adjectives and the polarities are sent into MOS Algorithm.

- The MOS algorithm takes 3 arguments as inputs
- The first argument is the list of adjectives which are used to express opinions (obtained from the 1st algorithm). These adjectives are given scores between [-4, +4] in such a way that a high score indicates a stronger opinion than lower score.

Example:

The adjective 'excellent' gets a higher positive opinion score than the adjective 'good'.

• The second argument is the list of inversion words like ‘not’ or 'never' which inverts the opinion orientation, so when these words occur in the left context of opinion words, they change the opinion sense.

• Example:

If the review is "The image quality is not good", then image quality must be given a negative score. So, these inversion words are kept track of.

• The third argument is the list of potential features obtained by using HAC algorithm.

• For each sentence in the review, we look at the opinion words and identify features closest to it. The score of feature is summation of scores of opinion words associated with it.

● The scores of features are further summed up to calculate score of review. So, for each review, a score is obtained and reviews are classified and ranked based on this score.

● A negative score for the review implies a negative review. The reviews of a particular category (+ve / -ve) are ranked based on their scores.

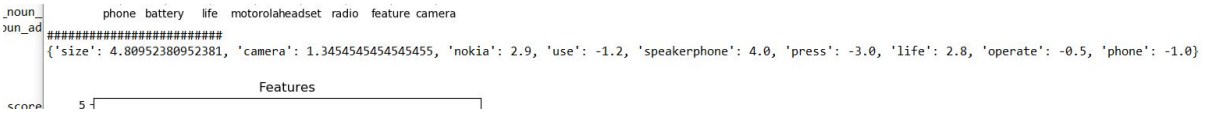

Figure 4.7 Scores of Features

Snapchat 4.7 Shows score of each feature from the dictionary.

4.2.5 Representing each Feature vs Score (Bar Graph):

Each potential feature and its score are plotted on Bar Graph

Potential Feature On X-Axis

Scores On Y-Axis

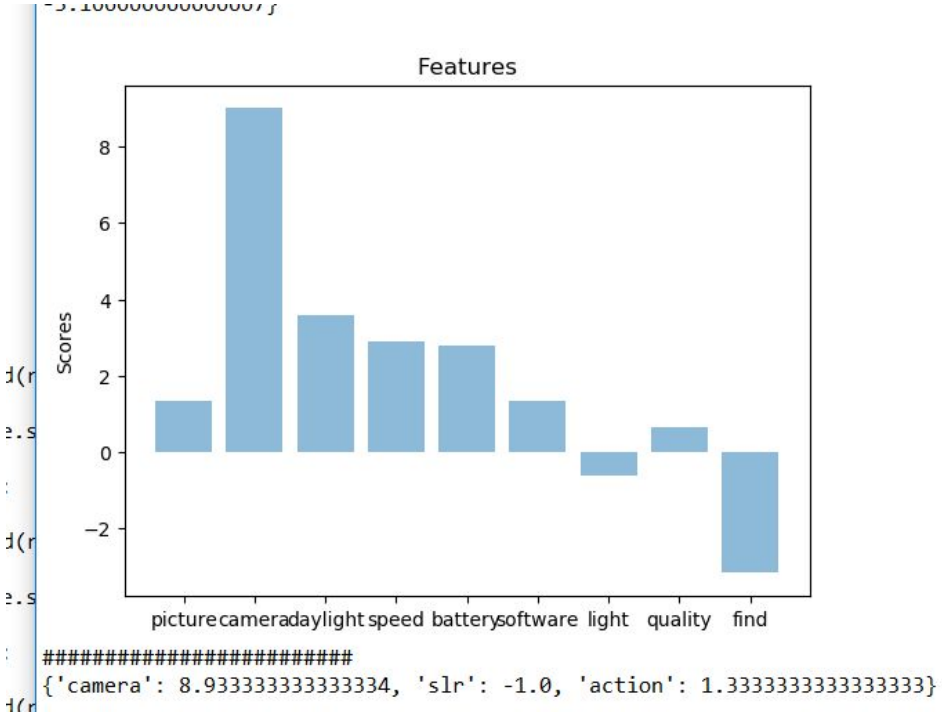

Figure 4.8 Output Example

Figure 4.8 shows bar-graph representation of features and its scores

TOOLS USED

DataSet Used:

This dataset consists of reviews from amazon.com

DataSet Format:

Example review of a mobile:

[+][t]excellent phone , excellent service .

##i am a business user who heavily depend on mobile service .

phone[+3], work[+2]##there is much which has been said in other reviews about the features of this phone , it is a great phone , mine worked without any problems right out of the box .

SOFTWARES:

- **Anaconda**

  Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. The Anaconda distribution is used by over 12 million users and includes more than 1400 popular data-science packages suitable for Windows, Linux, and MacOS.

- **Numpy**

  NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

- **NLTK**

  The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language. It was developed by Steven Bird and Edward Loper in the Department of Computer and Information Science at the University of Pennsylvania. NLTK includes graphical demonstrations and sample data. It is accompanied by a book that explains the underlying concepts behind the language processing tasks supported by the toolkit, plus a cookbook.

- **Matplotlib**

  Pyplot is a Matplotlib module which provides a MATLAB-like interface. Matplotlib is designed to be as usable as MATLAB, with the ability to use Python, and the advantage of being free and open-source.

- **TextBlob**

  TextBlob is a Python (2 and 3) library for processing textual data. It provides a consistent API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, and more.

- **Languages used: Python 3.7.0**

  Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural. It also has a comprehensive standard library.

- **System softwares: Windows 10**

  Windows 10 is a series of personal computer operating systems produced by Microsoft as part of its Windows NT family of operating systems. It is the successor to Windows 8.1, Windows 10 receives new builds on an ongoing basis, which are available at no additional cost to users, in addition to additional test builds of Windows 10 which are available to Windows Insiders. Devices in enterprise environments can receive these updates at a slower pace, or use long-term support milestones that only receive critical updates, such as security patches, over their ten-year lifespan of extended support

HARDWARES:

- Intel Core i5-7200u CPU @2.50GHz 2.70GHz
- 8.00GB RAM
- 64bit operating system, x64-based processor

IMPLEMENTATION

Here the implementation was about two main algorithms

1. HAC
2. MOS

PseudoCode for HighAdjectiveCount:

```
HighAdjectiveScore(reviews)
{
   Noun_score_map = dict()
   For each review in reviews do
       Assign part of speech tags to the review
       Apply stemming

  For each line in the review do
      For each adjective in the line find the closest noun
          Noun_score_map[noun]++

 potential _features_map = dict()

  If noun_score_map[noun] is greater than Threshold
     potential_features_map[noun]= Threshold

 Return potential_features_map

}
```

PseudoCode for MaximumOpinionScore(MOS) :

```
MOS(Adjective_Scores , inversionWords , potential_features ,
reviews)
{
    global_noun_scores= {}
    global_noun_adj_count={}

   For each review in reviews
      Review_noun_scores = {}
      Review_noun_adj_count ={}
      Apply POS Tagging and Stemming to review
  For each line in review
    For each word in line
       If word in adjective_scores
         Score = adjective_score[word]
      If inversion word on left side
        Score =-1*Score
  closest_noun=find_closest_noun(word)

  review_noun_score[closest_noun]+=Score
  review_noun_adjective_score[closest_noun]+=1
  global_review_noun_score[closest_noun]+=Score
  global_review_noun_adjective_score[closest_noun]+=1

Total_score= sum of all Scores in review_noun_score
Total_adjectives=      sum      of      all      Scores      in
review_noun_adjective_count
Avg_score =total_score/total_adjectives

If avg_score >0
   Mark review as positive
Else
  Mark review as negative

For each noun in global_noun_score

avg_feature_score[noun]=global_noun_score[noun]/global_noun_ad
jective_count[noun]
Rank Features by avg_feature_score
}
```

Following code will process the data set to the form of set used by all standard algorithms:

```
review=[];
reviewcontent=[];
reviewtitle=[];
Initially preprocessing the dataset
Removing the tags for every review
And appending it into reviewcontent list:
   Insert review into reviewContent List
reviewtitle=[]
Removing the tags in every review title
And appending it into the reviewtitle list
   Insert reviewtitle into reviewContent List
```

Following code will process the data by removing stop words and tagging is done and list containing nouns and adjectives is generated:

```
phrasesDict=OrderedDict(sorted(phrasesDict.items(),
key=operator.itemgetter(1), reverse=True))
#print(phrasesDict)
Create newPhrases dictionary()

for line_words, count in items in phrases dictionary:
          #Preprocessing text

    Removing the apostrophe_list words
    Remove ch for ch in line_words if ch not in exclude
     re.sub(r' [a-z][$]? ', ' ', line_words)
   Lemmatize the total review
   Updating the new phrases dictionary by checking
   Sorting in reverse order
   #print (newPhrases)

   nouns1 = []

  Appending the phrases having a value greater than threshold
  Pos_tagging
```

```
for a in range(length of reviewContent):
    Write "[t]"
    text=reviewTitle[a]
    Taging the text and store it in x variable
    #print (x)
    e=0
    while e <length of x:
     Creating the list containing only adjectives and nouns
           Write space
      Write ".\r\n"
```

Following code will take the input of feature_adjective_list and create the list of adjective defining feature and set of nouns:

```
##hac

HighAdjectiveScore(reviews)
{
   Noun_score_map = dict()
   For each review in reviews do
       Assign part of speech tags to the review
       Apply stemming

  For each line in the review do
      For each adjective in the line find the closest noun
          Noun_score_map[noun]++

 potential _features_map = dict()

  If noun_score_map[noun] is greater than Threshold
     potential_features_map[noun]= Threshold

 Return potential_features_map

} For each noun in noun_score_map
```

Following code will create adjective scores:

```
Initiate Adjective scores dictionary to store scores of each
individual adjectives
Iterate through adjective list:
    Using text blob library get the polarity of each adjective
    If polarity obtained is not zero :
    Add them into adjective scores dictionary

Iterate through the adjective score list:
 Update each score on scale of -4 to 4 by multiplying polarity
 with 4
adj scores
```

Negation before particular adjective can be find by the following steps:

```
If word is present in adjscores dictionary then:

   Store score in score variable

   If wordindex is greater or equal to 2 then
      Phrase is created by concatenating previous two words
      Now check the polarity of the phrase
      If polarity is less than zero then:
         Multiply the total score with -1

   If wordindex is greater or equal to 1 then
      Phrase is created by concatenating previous word
      Now check the polarity of the phrase
      If polarity is less than zero then:
         Multiply the total score with -1
```

And the score is calculated in mos by using the below steps:

```
Store closest Noun in closest_noun variable by  user defined
function with parameters as feature list and current word
index
```

```
If Closest Noun is not found:
    Then continue
If closest noun is found :
    Then
    If closest noun is present in noun scores dictionary:
       Then
       Increase the closest noun dictionary value by score
    Else
       Insert closest noun into noun score dictionary and
       initiate its value as score

   If closest noun is present in noun adjective count
   dictionary:
       Then
       Increase the closest noun dictionary value by one
    Else
       Insert closest noun into noun score dictionary and
       initiate its value as one

    If closest noun is present in Global noun scores
    dictionary:
       Then
       Increase the closest global noun dictionary value by
       score
    Else
       Insert closest noun into global noun score dictionary
       and initiate its value as score

   If closest noun is present in global noun adjective count
   dictionary:
       Then
       Increase the closest global noun dictionary value by
       one
    Else
       Insert closest noun into global noun score dictionary
       and  initiate its value as one
```

And for plotting using bar graphs:

```
import matplotlib.pyplot as plt; plt.rcdefaults()
```

```
import numpy as np
import matplotlib.pyplot as plt


y_pos = np.arange(len(objects))

plt.bar(y_pos, performance, align='center', alpha=0.5)
plt.xticks(y_pos, objects)
plt.ylabel('Score')
plt.title('Features')

plt.show()
```

RESULT ANALYSIS

The final result displayed was about the feature and its score. It was displayed in the range of -4 to 4. The normal of feature score was as below diagram

And the bar-graph was the final pictorial representation of the total score which can be negative also.

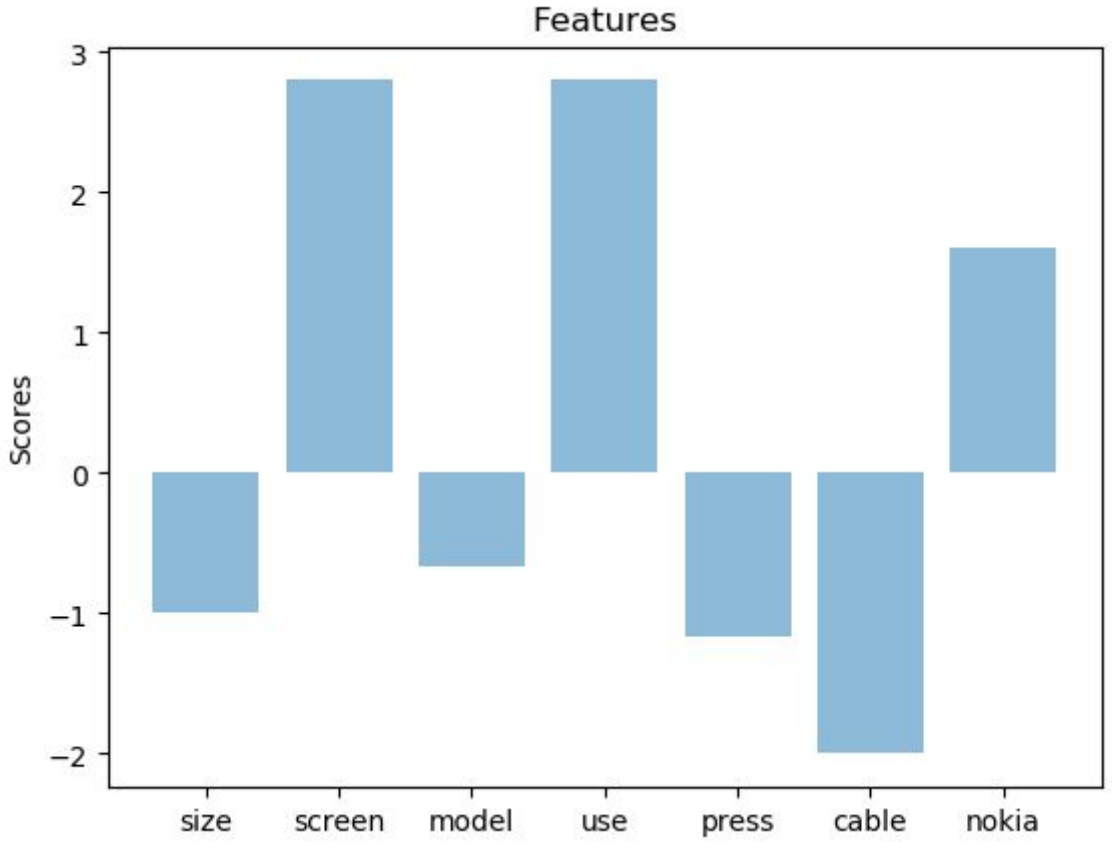

Figure 7.1 Representing the Features and its Scores in Bar-Graph

More the Score the more it was liked by users.Negative score implies majority users don't like the specific feature of the product.

CONCLUSION & FUTURE SCOPE

In this work, we have presented a method to extract product features from customer product reviews. The objective is to provide a feature based opinion of a large number of customer reviews of a particular product. The problem, analysis of customer product reviews, will become increasingly important as more people are buying and expressing their opinions on the Web.Analyzing the opinions of reviews is not only useful to common shoppers, but also crucial to product manufacturers.

Product feature extraction can be further improved by adding implicit feature extraction. Some review sentences only include sentiment words without associated feature terms.The product features that do not appear in review sentences but are actually referred to are called implicit features.this can be acheived by using deep learning which was step next to the work done here

REFERENCES

[1] D. Winder, The importance of social mobility, Gemalto Review Magazine (2010) .

[2] V. Hatzivassiloglou, K. R. McKeown, Predicting the semantic orientation of adjectives, in: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, ACL '98, 1997, pp. 174–181.

[3] P. D. Turney, M. L. Littman, Measuring praise and criticism: Inference of semantic orientation from association, ACM Trans. Inf. Syst. 21 (2003) 315–346.

[4] H. Kanayama, T. Nasukawa, Fully automatic lexicon expansion for domain-oriented sentiment analysis, in: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP '06, 2006, pp. 355–363.

[5] B. Pang, L. Lee, Opinion mining and sentiment analysis, Found. Trends Inf. Retr. 2 (2008) 1–135.

[6] A. Esuli, F. Sebastiani, Determining the semantic orientation of terms through gloss classification, in: Proceedings of the 14th ACM international conference on Information and knowledge management, CIKM '05, 2005, pp. 617–624.

[7] S.-M. Kim, E. Hovy, Determining the sentiment of opinions, in: Proceedings of the 20th international conference on Computational Linguistics, COLING '04, 2004.

[8] A. Meena, T. V. Prabhakar, Sentence level sentiment analysis in the presence of conjuncts using linguistic analysis, in: Proceedings of the 29th European conference on IR research, ECIR'07, 2007.

[9] T. Nasukawa, J. Yi, Sentiment analysis: capturing favorability using natural language processing, in: Proceedings of the 2nd international conference on Knowledge capture, K-CAP '03, 2003, pp. 70–77.

[10] Deptii D. Chaudhari, Prof. R.A. Deshmukh et.al. “Feature Based Approach for Review Mining Using Appraisal Words”, 2013 International Conference on Emerging Trends in Communication, Control, Signal Processing and Computing Applications (C2SPCA),

[11] ”Natural Language Processing (NLP) Techniques for Extracting Information”

[12] ”A review of natural language processing techniques for opinion mining systems**”**

[13] B. Pang, L. Lee. ”Opinion mining and sentiment analysis”. Found Trends Inf. Retrieval, 2 (2008), pp. 1-135

[14] J.A. Balazs, J.D. Velásquez. ”Opinion mining and information fusion: a survey” Inf.