Product Feature Ranking and Review Classification

A PROJECT REPORT

Submitted by

HARI KRISHNAN K (715517104030)

DINESH KUMAAR K (715517104302)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

PSG INSTITUTE OF TECHNOLOGY AND APPLIED

RESEARCH, COIMBATORE 641 062

ANNA UNIVERSITY: CHENNAI - 600 025

JUNE 2021

ANNA UNIVERSITY: CHENNAI - 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"PRODUCT FEATURE RANKING AND REVIEW CLASSIFICATION"** is the bonafide work of **"HARI KRISHNAN K (715515104030), DINESH KUMAAR K (715515104302)"** who carried out the project work under my supervision.

[illegible]

SIGNATURE

Dr. R. Manimegalai

HEAD OF THE DEPARTMENT

Computer Science and Engineering

PSG Institute of Technology and

Applied Research,

Coimbatore – 641 062

[illegible]

SIGNATURE

Ms. K Lakshmi Kalpana Roy

SUPERVISOR

Assistant Professor (Sr. Gr.)

Computer Science and

Engineering

PSG Institute of

Technology and Applied

Research,

Coimbatore – 641 062

Submitted for the project viva-voce Examination held on__________

---------------------------------- **----------------------------------**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENTS

We are very thankful to **Shri L Gopalakrishnan,** Management Trustee, PSG & Sons' Charities for providing me with sufficient resources for completing out project.

We are very grateful to **Dr G Chandramohan,** Principal and Professor for providing me with an environment to complete our project successfully.

We are greatly indebted to **Dr P V Mohanram**, Secretary and Professor for his constant encouragement and motivation.

We express my sincere thanks to **Dr R Manimegalai**, Head of the Department and Professor for her instrumental guidance and support throughout the project.

We sincerely thank my Project Guide **Ms. K Lakshmi Kalpana Roy,** Assistant Professor (Senior Grade) for her consistent support throughout the course.

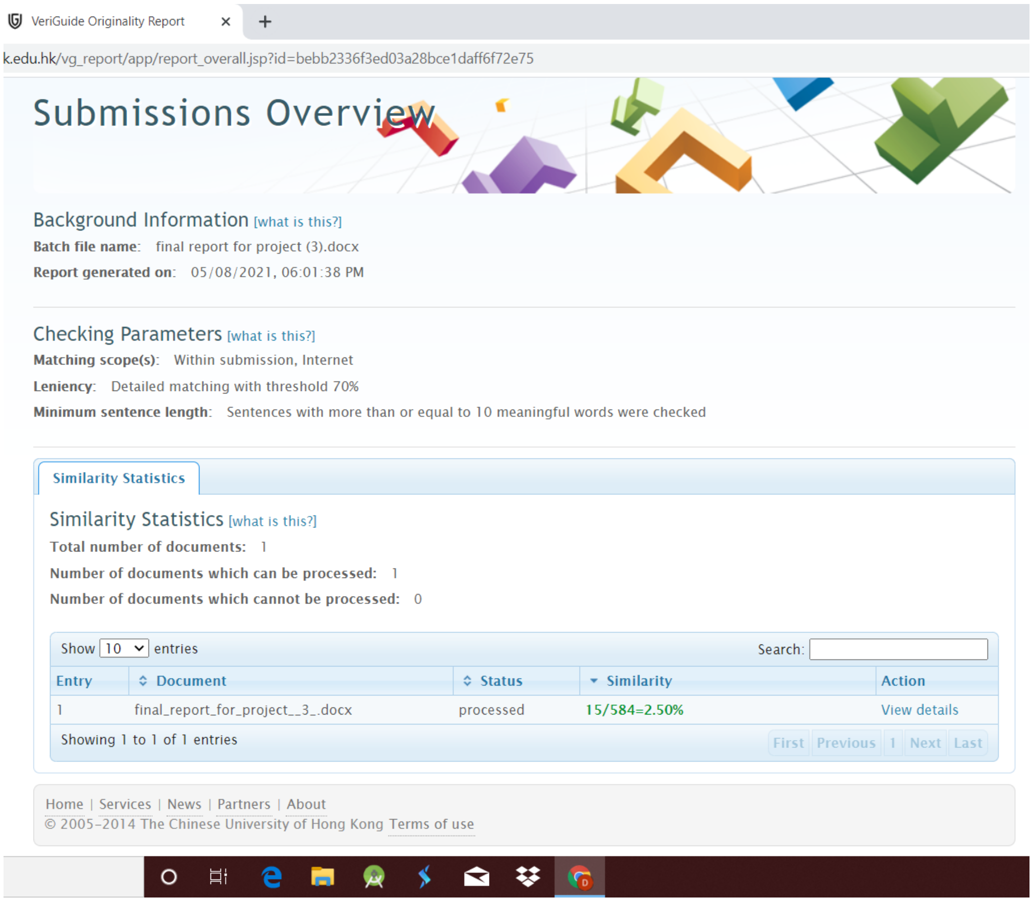
We earnestly thank my Project Coordinator **Mr. S Thivaharan,** Assistant Professor (Selection Grade) for his guidance throughout the project.

Finally, we take this opportunity to extend my deepest appreciation to our **family and friends**, who supported us during the crucial times of our project.

HARI KRISHNAN K

DINESH KUMAAR K

VERIGUIDE ANALYSIS REPORT



VeriGuide Originality Report

k.edu.hk/vg_report/app/report_overall.jsp?id=bebb2336f3ed03a28bce1daff6f72e75

Submissions Overview

Background Information [what is this?]

Batch file name: final report for project (3).docx

Report generated on: 05/08/2021, 06:01:38 PM

Checking Parameters [what is this?]

Matching scope(s): Within submission, Internet

Leniency: Detailed matching with threshold 70%

Minimum sentence length: Sentences with more than or equal to 10 meaningful words were checked

Similarity Statistics

Similarity Statistics [what is this?]

Total number of documents: 1

Number of documents which can be processed: 1

Number of documents which cannot be processed: 0

Show 10 entries   Search:

Entry	Document	Status	Similarity	Action
1	final_report_for_project__3_.docx	processed	15/584=2.50%	View details

Showing 1 to 1 of 1 entries   First Previous 1 Next Last

Home | Services | News | Partners | About

© 2005-2014 The Chinese University of Hong Kong Terms of use

ABSTRACT

The e-commerce web sites are designed in a way for their customers or clients to share their reviews and comments on the products through ratings. The ratings provide a sign about the where the products stand in the market while at the same time provide an opinion about the products when customers view the website for purchasing a product. Since the evolution of social networks, people have started to express their opinions in the form of the blogs or facebook posts or tweets starting from the products people buy to the presidential candidate they support. When searched for a particular product on the web, the current day search engines show the list of websites which gives the features of the product and their prices. But, the users can be given much more info about the product using the reviews about the searched product. Thus, a new type of module can be designed which will not only retrieve facts, but will also enable the retrieval of opinions of the users about the product. We create a module which, when searched for a product, gives the highlighted features of the product and score and some of the helpful reviews (instead of the user scrolling through all the reviews to know about the product). This project proposes a system that provides the users an authentication of any product they wish, based on the online reviews and comments posted by the customers who have already had an experience of it. It aims to provide summarized positive and negative features about products and services by analyzing their online reviews platform is available. Neutral and veracious reviews help online users in making a right decision for themselves and obtaining the right product suited for them. Reviews act as a trust-building mechanism and creates a good relationship between the involved entities.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABBREVIATION	**EXPANSION**
AI	ARTIFICIAL INTELLIGENCE
ML	MACHINE LEARNING
NLP	NATURAL LANGUAGE PROCESSING
NLTK	NATURAL LANGUAGE PROCESSING TOOLKIT
SVM	SUPPORT VECTOR MACHINE
HAC	HIGH ADJECTIVE SCORE
MOS	MAXIMUM OPINION SCORE

CHAPTER 1

INTRODUCTION

1.1 E-COMMERCE

Electronic commerce or e-commerce (sometimes written as eCommerce) is a business model that lets firms and individuals buy and sell things over the internet.E-commerce, which can be conducted over computers, tablets, or smartphones may be thought of like a digital version of mail-order catalog shopping. Nearly every imaginable product and service is available through e-commerce transactions, including books, music, plane tickets, and financial services such as stock investing and online banking. As such, it is considered a very disruptive technology.Major E-commerce websites used in India shown below in figure 1.1.1

Top E-Commerce Websites in India
MYNTRA.com
India's Largest Online Fashion Store
snapdeal.com
JABONG.COM
amazon.in
flipkart.com

Figure 1.1: E-com websites

The Indian e-commerce scene has a nice mix between pure e-commerce players and brick, and mortar retailers turned e-commerce giants. While the Indian e-commerce sector is still in early development stages, it is a market rich with opportunity; with so many people comes so much room for major players and small players alike to take a giant chunk out of the Indian e-commerce pie.

Top 10 Ecommerce Sites in India 2020

- Amazon India
- Flipkart
- Alibaba
- Snapdeal
- Myntra
- IndiaMART
- Book My Show
- Nykaa
- Firstcry
- 1mg

American e-commerce giant, Amazon, is said to have an audience reach of 89 percent in India, according to Statista. Since launching in India in 2010, the site now generates an estimated 322.54 million monthly visitors, making it the highest performing site in the country, by a long shot. True to the overall statistics that the primary e-commerce category in India is electronics.When searched for a particular product on the web, the current day search engines show the list of websites which gives the features of the product and their prices.So we create a module which, when searched for a product, gives the highlighted features of the product and score and some of the helpful reviews [19].

1.2 OBJECTIVE

The main objective of the project is to provide the customer the easier understanding about the product. The user who want to buy a product did not need to study the whole review to know about the product. By classifying product reviews into positive, negative and neutral the customer can read reviews according to his choice. By giving score to features for the product the customer can easily get a clear idea about the product.

1.3 PROBLEM STATEMENT

The features of the product is the important thing which make customers buy it. Each product has its own advantage and disadvantages. Also most of the products contains same features. But what make difference is build quality or quality of service. By which the product becomes brand. By giving feature score based on customer review one who wants to buy product can easily know about and he may get a clear conclusion that it will satisfy his requirements or not.

1.4 PROJECT OVERVIEW

The product reviews are first get collected from online shopping sites and then the collected reviews are get saved in text file. The text files are the datasets. Then the collected datasets are pre-processed and then used for further process.

Basically, two algorithms are used for the purpose:

The 1st algorithm (HAC) identifies and extracts the potential features from the reviews of the product.

The 2nd algorithm takes these potential features as input, assigns scores to them and finally helps in classifying every review as positive, negative or neutral. The scores obtained for every feature of the product can be used to highlight the good features of the product.

1.5 SEMANTIC ANALYSIS

Sentimental analysis is the sector of comprehending feelings and expression of humans which they express within the form of critiques, feedback, and guidelines. In this point in time, humans have the liberty to express themselves on numerous structures, that allow them to hook up with a tremendous majority of the network. those may be shared via various strategies, like discussion boards, YouTube channels, evaluations blogs, and social media posts. These facts may be shared by using every person, which does not necessarily, ought to be professionals of the sphere. The evaluation of this facts in a condensed manner can be classified into positive, bad or neutral in nature. And it may additionally be categorised primarily based on various variables. Such freedom to express, may additionally lead to fake statistics being shared around.

This facilitates in producing false hype and endorsement of a private product schedule driven critiques end up a not unusual problem when any such platform is to be had. From producers to shops, absolutely everyone would possibly try to perplex potential clients in buying their product impartial and veracious evaluations assist online users in creating a right choice for themselves and acquiring the right product ideal for them. A correct analysis allows every party involved on this precise on the spot. The opinions act as a consider-building mechanism and creates a good courting between the involved entities. In gadget gaining knowledge of, category is used to classify a new statement into a specific set/class based totally on a training set of information containing observations whose class is thought earlier.

The maximum commonplace example is “junk mail” or “non-junk mail” training for emails. In E-trade, classifier algorithms can be used to classify sentiments of evaluation primarily based on phrases. The particular phrases within the language are categorised earlier for their high-quality or poor sentiments. Mastering training set has efficaciously recognized observations. Classifier algorithms are used to create cluster/units from the uncategorized unsupervised records primarily based on similarity and/or distance from the schooling facts set.

1.6 NATURAL LANGUAGE PROCESSING

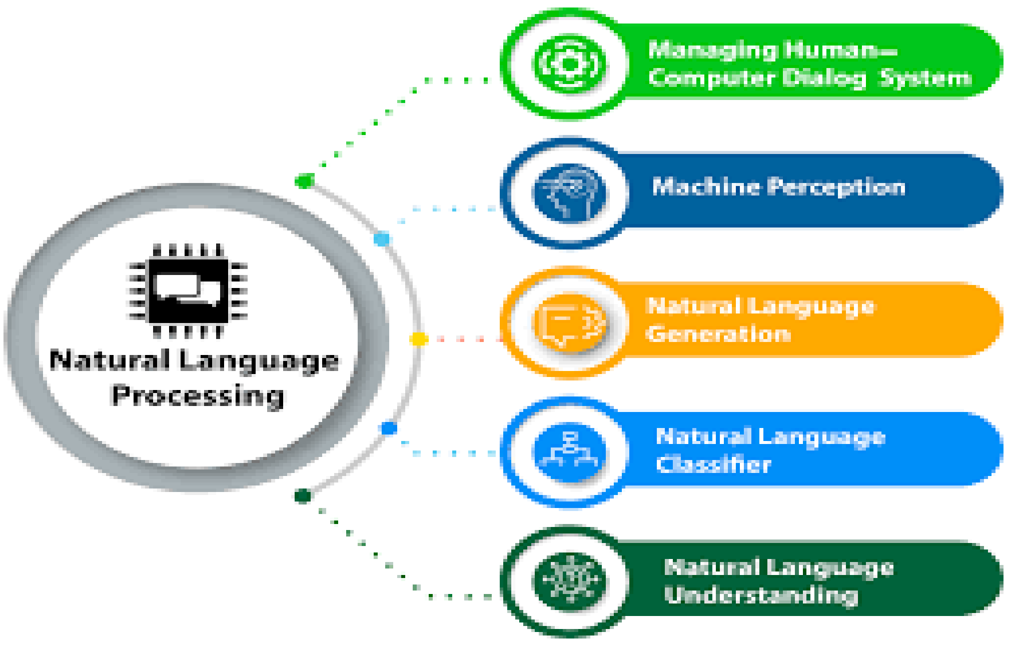

Natural Language Processing
Managing Human–Computer Dialog System
Machine Perception
Natural Language Generation
Natural Language Classifier
Natural Language Understanding

Figure 1.2: NLP

Natural language processing (NLP) is the intersection of computer technology, linguistics natural language and NLP is all approximately making computer systems apprehend and generate human language programs of NLP strategies include voice assistants like Amazon's Alexa and Apple's Siri, but additionally such things as machine translation and textual content filtering. NLP has closely benefited from current advances in machine learning, specially from deep learning techniques. The sphere is split into the three components: Speech popularity,the translation of spoken language into text. Natural Language technology. The era of natural language with the aid of a computer scientific improvements in NLP can be divided into three classes (Rule-based systems, Classical machine learning models and Deep learning models).

Rule-based systems totally rely closely on crafting area-particular regulations (e.g., everyday expressions), can be used to solve easy problems inclusive of extracting dependent records (e.g., emails) from unstructured information (e.g., web-pages), but due to the complexity of natural human languages, rule-based totally structures fail to build models that can purpose about language.

Classical machine learning strategies may be used to solve extra hard troubles which rule-primarily based systems can't solve very well (e.g., spam Detection), it relies on an extra well known approach to understanding language, the usage of hand crafted capabilities (e.g., sentence length, part of speech tags, the incidence of precise words) then supplying the ones functions to a statistical device mastering version (e.g., Naive Bayes), which learns exceptional patterns in the schooling set after which be able to purpose approximately unseen records (inference).

Deep learning models are the hottest a part of NLP research and applications now. They generalize even better than the classical machine knowledge of processes as they don’t want hand crafted functions because they routinely works as feature extractors, which helped lots in building end to quit fashions (little human-interplay) aside from the feature engineering part, deep learning algorithms getting to know abilities are more effective than the shallow/classical ML ones, which paved its manner to attaining the highest scores on different hard NLP tasks (e.g., Machine Translation)[21].

1.7 NATURAL LANGUAGE PROCESSING IN OPINION MINING

One of the forms of natural language processing is opinion mining which offers with tracking the mood of the humans concerning a specific product or topic. This software program presents computerized extraction of critiques, emotions and sentiments in text and additionally tracks attitudes and emotions on the internet. People express their views by means of writing blog posts, feedback, critiques and tweets about all types of unique subjects. tracking merchandise and types and then figuring out whether or not they're regarded positively or negatively may be performed using web.

The opinion mining has slightly exceptional tasks and plenty of names, e.g. sentiment evaluation, opinion extraction, sentiment mining, subjectivity evaluation, affect evaluation, emotion evaluation, evaluate mining and many others. But, all of them come under the umbrella of sentiment evaluation or opinion mining. Sentiment category, feature based totally sentiment category and opinion summarization are few major fields of research predominate in sentiment analysis.

In recent years, the system got witnessed that supporting postings in social media have helped reshape groups, and sway public sentiments and feelings, that have profoundly impacted on our social structures. It has hence emerged as a necessity to gather and observe evaluation at the web. Of route, opinionated files no longer only exist at the net (referred to as external facts), many corporations additionally have internal information, e.g., consumer comments collected from emails and phone centres or consequences from surveys conducted by using the corporations. Opinion mining may be useful in several ways. For example, in marketing, it tracks and judges the fulfillment price of an advert marketing campaign or launch of new product, decide popularity of products and services with its versions additionally inform us approximately demographics which like or dislike unique features[22].

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEMS

The number of papers dealing with literature is growing exponentially. Several researchers have played a significant role in the development of useful opinion mining and product review classification algorithms.

K. L. S. Kumar, J. Desai and J. Majumdar[1] concentrated on classifying online reviews that have been extracted from Amazon by using different algorithms. These algorithms that include Naive Bayes Classifier, Logistic Regression and SentiWordNet algorithm help in determining the polarity of the online reviews.

Z. Singla, S. Randhawa and S. Jain [2] focused on emotional sentiment analysis which not only will make it possible to classify the reviews as positive or negative, but also to classify their based on the emotion which can be of anger, fear, happiness, disappointment, etc.

P. P. Surya and B. Subbulakshmi [3] have used Naïve Bayes Classifier for the polarity classification and analysis using R tool which is a data mining tool. The outcome of the process is presented in the form of a confusion matrix.

Abinash Tripathya, Ankit Agrawalb[4],Support Vector Machine (SVM) technique has been used by the authors for extracting the polarity of a review. Analysis of frequency, precision and recall has been done by the authors for the proposed algorithm. The Support Vector Machine (SVM) technique outperforms in every case.

M. S. Neethu and R. Rajasree[5],When extracting online reviews, one of the major concerns is of handling slang words or misspelled words. Regarding this problem, developed a feature extraction procedure which involved vectorization and preprocessing of the text. Product reviews can be used to extract whether specific features of the product are good or bad according to the customer.

A. Agarwal, V. Sharma[6] carried out an opinion miner which was feature based. The main work of the miner is to find the significant features of the product by investigating the review and making the assessment profile of every product which can be utilized by the user.

Y. Li, X. Feng and S. Zhang[7] have exhibited three kinds of new features which incorporate review density, semantic and emotions. The authors have provided the model and algorithm to build each feature. They have concluded that the proposed model, calculation and features are productive in fake review detection process. Regarding fake review detection the authors justify that the utilization of ratings alone to determine whether the review is phony or fake is insufficient, as the data that can be extracted is restricted.

2.2 DRAWBACKS IN THE EXISTING SYSTEM

- The systems involves complex processing methods.
- The size of the software is usually large and it isn’t compatible with all the systems.
- The systems does not give score for features.
- A generalised approach is not seen in any of the above projects.

CHAPTER 3

SYSTEM DESCRIPTION

The system's algorithm is written in python and the project runs as an application embedded with a frontend developed using HTML, CSS and JS. The application runs in local IP and can also be deployed in an instance of a cloud or a DNS to run at a particular IP address. The software used in the system are described in the following sections.

3.1 NATURAL LANGUAGE TOOLKIT

NLTK is a main platform for constructing Python programs to work with human language records. It affords easy-to-use interfaces to over 50 corpora and lexical resources along with WordNet, along with a suite of textual content processing libraries for type, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial strength NLP libraries, and an energetic dialogue discussion board. NLTK has been referred to as "a outstanding tool for coaching, and operating in, computational linguistics using Python," and "an terrific library to play with herbal language". NLTK is suitable for linguists, engineers, college students, educators, researchers, and industry customers alike. NLTK is available for home windows, Mac OS X, and Linux and it is a loose, open supply, community-driven assignment[23].

3.2 NUMPY

1. NumPy is the fundamental bundle for scientific computing in Python. it is a Python library that gives a multidimensional array object, diverse derived items (together with masked arrays and matrices), an

collection of routines for instant operations on arrays which includes mathematical, logical, shape manipulation, sorting, choosing, I/O, discrete Fourier transforms, linear algebra, statistical operations, random simulation and lots greater. On the core of the NumPy package, is the *nd*array object. This encapsulates n-dimensional arrays of homogeneous information types, with many operations being done in compiled code for performance.

3.3 PYENCHANT

Pyenchant is a spellchecking library for Python, primarily based on the splendid Enchant library. Pyenchant combines all of the capability of the underlying Enchant library with the power of Python and a pleasant “Pythonic” object-orientated interface. Pyenchant 3.2.0 Enchant is used to check the spelling of words and endorse corrections for phrases which are miss-spelled. it may use many famous spellchecking applications to carry out this undertaking, which include ispell, aspell and MySpell.

3.4 SIX

Six is a Python 2 and 3 compatibility library. It affords software capabilities for smoothing over the differences between the Python variations with the goal of writing Python code this is well matched on each Python variations. Six provides easy utilities for wrapping over differences among Python 2 and Python three. It's far meant to support codebases that work on both Python 2 and 3 without modification. Six consists of most effective one Python record, so it's miles painless to replicate into a venture. The name, “six”, comes from the fact that 2*three equals 6. Multiplication is more effective, and, besides, “5” has already been snatched away via the Zope 5 mission.

3.5 TEXT BLOB

TextBlob is a Python (2 and 3) library for processing textual facts. It presents a simple API for diving into not unusual natural language processing (NLP) obligations which includes element-of-speech tagging, noun word extraction, sentiment analysis, class, translation, and extra.

3.6 TEXT TABLE

It is a python module, which allows us to print table on terminal. it's miles one of the basic python modules for reading and writing textual content tables in ASCII code. It pursuits to make the interface as similar as feasible like csv module in Python.

3.7 NLTK DATA

The NLTK records module includes features that may be used to load NLTK aid files, inclusive of corpora, grammars, and saved processing items. Other than man or woman facts applications, the complete series (the usage of “all”), or just the facts required for the examples and exercises within the e book (using “e-book”), or simply the corpora and no grammars or skilled fashions (using “all-corpora”) can be downloaded.

CHAPTER 4

SYSTEM DESIGN

4.1 PROPOSED SYSTEM

The system proposed to minimize the users time to understand about the product and to give a easiest possible understanding about the product and its features. Providing ranks and scores for products and its features provides clear vision about product in short time and also don't need to study about lengthy reviews.

4.2 COLLECTION OF DATA

The dataset for this project has been collected from various open-source websites and shopping sites. The dataset is the pre-processed. to make the system processing fast unwanted words are removed from the reviews contained in text file. Adjectives and features are filtered out and scores are given for feature using algorithms. The datasets are nothing but text files which contain reviews.

4.2.1 ATTRIBUTES REQUIRED FOR PROCESSING

- **Adjectives:** The adjectives is used to find whether the review is positive or negative and it is also used to give scores.
- **Nouns:** The nouns are the features for which the score is given based on adjective count.

- **Inversion words:** These are the words which are associated with negations like not, no, etc.
- The inversion words is used to find that the review is negative and also again it is an adjective too. So if more inversion words are related to noun or feature the scores will get reduce.
- **Scores:** The score range from [-4 to 4] which are given to features based on adjectives count.

4.3 MODEL DEVELOPMENT

The model is developed using Natural language processing. NLP is an multi-disciplinary area of artificial intelligence. In this project linguistics is handled because the data sets are reviews which is human language. Natural language toolkit package is used to process the reviews. Natural language toolkit data consists of packages like chunkers, corpora, grammars, etc. By using algorithms potential features from reviews are extracted and then reviews are classified as positive, negative and neutral and then the scores for features are given using algorithms.

4.4 NATURAL LANGUAGE PROCESSING USED

The NLP utilized in this project is semantic evaluation. Semantic evaluation describes the method of know-how natural language–the manner that people communicate–based on meaning and context. It allows to study how a cognitive technology like expert plays semantic analysis. The semantic evaluation of natural language content begins with the aid of analyzing all of the phrases in content to capture the actual that means of any textual content. It identifies the text elements and assigns them to their logical and grammatical function. It analyzes context in the surrounding textual content and also the text shape to correctly disambiguate the proper meaning of words which have a couple of definition. Semantic era procedures the logical shape of sentences to discover the maximum relevant factors in text and recognize the topic discussed. It additionally is familiar with the relationships between extraordinary concepts within the textual content. For example, it knows that a textual content is about "politics" and "economics" despite the fact that it doesn't contain the real words but relate concepts consisting of "election," "Democrat," "speaker of the house", or "budget," "tax" or "inflation". Because semantic analysis and natural language processing can help machines robotically understand textual content, this supports the even larger aim of translating facts–that potentially treasured piece of patron remarks or insight in a tweet or in a customer service log–into the area of commercial enterprise intelligence for customer support, corporate intelligence or understanding management.

4.5 SEMANTIC ANALYSIS

Semantic evaluation is the technique of drawing that means from text. It allows computer systems to recognize and interpret sentences, paragraphs, or complete files, by way of analyzing their grammatical shape, and figuring out relationships between words in a particular context. Its an important sub-project of natural Language Processing (NLP) and the riding force at the back of machine learning tools like chatbots, serps, and textual content analysis. Semantic analysis-driven tools can assist groups robotically extract meaningful data from unstructured records, which include emails, guide tickets, and patron remarks[25].

4.5.1 WORKING OF SEMANTIC ANALYSIS

Lexical semantics plays an important role in semantic evaluation, permitting machines to understand relationships between lexical gadgets (phrases, phrasal verbs, and so forth.):

Hyponyms: Precise lexical gadgets of a established lexical item (hypernym) e.g. orange is a hyponym of fruit (hypernym).

Meronomy: A logical arrangement of text and words that denotes a constituent part of or member of something e.g., a segment of an orange

Polysemy: A relationship among the meanings of phrases or terms, although barely unique, percentage a not unusual middle meaning e.g. I read a paper, and that i wrote a paper)

Synonyms: Words that have the equal feel or almost the same which means as some other, e.g., glad, content, ecstatic, extremely joyful

Antonyms: Phrases that have near contrary meanings e.g., satisfied, sad

Homonyms: Two phrases that are sound the same and are spelled alike but have a unique meaning e.g., orange (color), orange (fruit)

Semantic analysis also takes under consideration signs and emblems (semiotics) and collocations (words that frequently cross together). computerized semantic evaluation works with the help of machine learning algorithms. By using feeding semantically greater device getting to know algorithms with samples of text, you can teach machines to make accurate predictions based on past observations.

4.5.2 SEMANTIC ANALYSIS TECHNIQUES

Relying at the form of information you'd like to attain from information, you may use one of two semantic evaluation techniques: a text type model (which assigns predefined categories to textual content) or a text extractor (which attracts out particular data from the text).

Semantic classification models

 Topic category: sorting text into predefined classes primarily based on its content material. Customer service teams may additionally want to categorise aid tickets as they drop into their help desk. Through semantic evaluation, device learning equipment can understand if a ticket should be categorized as a "payment trouble" or a "transport trouble."

Sentiment analysis: detecting nice, terrible, or neutral feelings in a textual content to indicate urgency. For instance, tagging Twitter mentions through sentiment to get a feel of ways clients experience approximately your brand, and being able to discover disgruntled customers in actual time.

Intent classification: classifying text based on what customers want to do next. This is used to tag income emails as "involved" and "no longer involved" to proactively attain out to folks that can also need to attempt your product.

Semantic Extraction models

Keyword extraction: locating relevant phrases and expressions in a textual content. This approach is used alone or alongside one of the above strategies to gain extra granular insights. As an instance, you could examine the keywords in a group of tweets that have been categorized as “negative” and stumble on which phrases or subjects are cited most customarily.

Entity extraction: figuring out named entities in text, like names of people, groups, places, and many others. A customer service team would possibly locate this useful to automatically extract names of products, shipping numbers, emails, and some other applicable facts from customer service tickets.

Mechanically classifying tickets the usage of semantic analysis equipment alleviates retailers from repetitive obligations and allows them to attention on duties that provide more cost at the same time as improving the complete purchaser experience. Tickets can be instantly routed to the right palms, and pressing issues may be without problems prioritized, shortening reaction times, and keeping delight stages excessive. Insights derived from information also help teams detect regions of improvement and make better choices. For example, you may determine to create a robust knowledge base by means of figuring out the maximum commonplace patron inquiries [24].

4.6 MODEL ALGORITHM

Basically, two algorithms are used for purpose:

- The 1st algorithm (HAC) identifies and extracts the potential features from the reviews of the product.
- The 2nd algorithm takes these potential features as input, assigns scores to them and finally helps in classifying every review as positive, negative or neutral. The scores obtained for every feature of the product can be used to highlight the good features of the product.

4.6.1 THE HIGH ADJECTIVE COUNT ALGORITHM

- Instead of using frequency of keywords, the algorithm starts identifying adjectives and nouns.
- The scores of nouns are initialized to zero, each adjective is associated with a noun to which it is closest, this adjective is more likely to describe the noun.
- For each such adjective, score of noun is increased by one. After processing all the reviews, will have a score associated with each noun, which will be then called as opinion scores.
- So nouns with high score have more adjectives to describe them. Then will have a threshold and nouns having score more than threshold are considered as potential features.

4.6.2 MAX OPINION SCORE ALGORITHM

This algorithm takes 3 arguments as input:

- The first argument is the list of adjectives which are used to express opinions,we refer them as opinion words. Have to chose a value manually between [-4,4] to each opinion word. A high score indicates a stronger opinion than lower score.
- The second argument is the list of inversion words like 'not' which give a negative sense to opinion,so when these words occur in the left context of opinion words,they change the opinion sense. So when a inversion word appears, then need to multiply the score by -1.
- The third argument is the list of potential features obtained by using TF / TF-IDF / HAC algorithm.

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 SYSTEM FLOW

The product reviews are first get collected from online shopping sites and then the collected reviews are get saved in text file. The text files are the datasets. Then the collected datasets are pre-processed and then used for further process. To make the system processing fast unwanted words are removed from the reviews contained in text file. Adjectives and features are filtered out and scores are given for feature using algorithms. The datasets are nothing but text files which contain reviews. Basically, two algorithms are used for the purpose: The first algorithm (HAC) identifies and extracts the potential features from the reviews of the product. The second algorithm takes these potential features as input, assigns scores to them and finally helps in classifying every review as positive, negative or neutral. The scores obtained for every feature of the product can be used to highlight the good features of the product. The adjectives can be used to find whether the review is positive or negative and it is also used to give scores. Here the nouns are the features for which the score is given based on adjective count. The words which are associated with negations like not, no etc,. The inversion words can be used to find whether the review is negative and also again it is an adjective too. So if more inversion words are related to noun or feature the scores will get reduce.

The figure 5.1 below shows a simple representation of the system architecture.

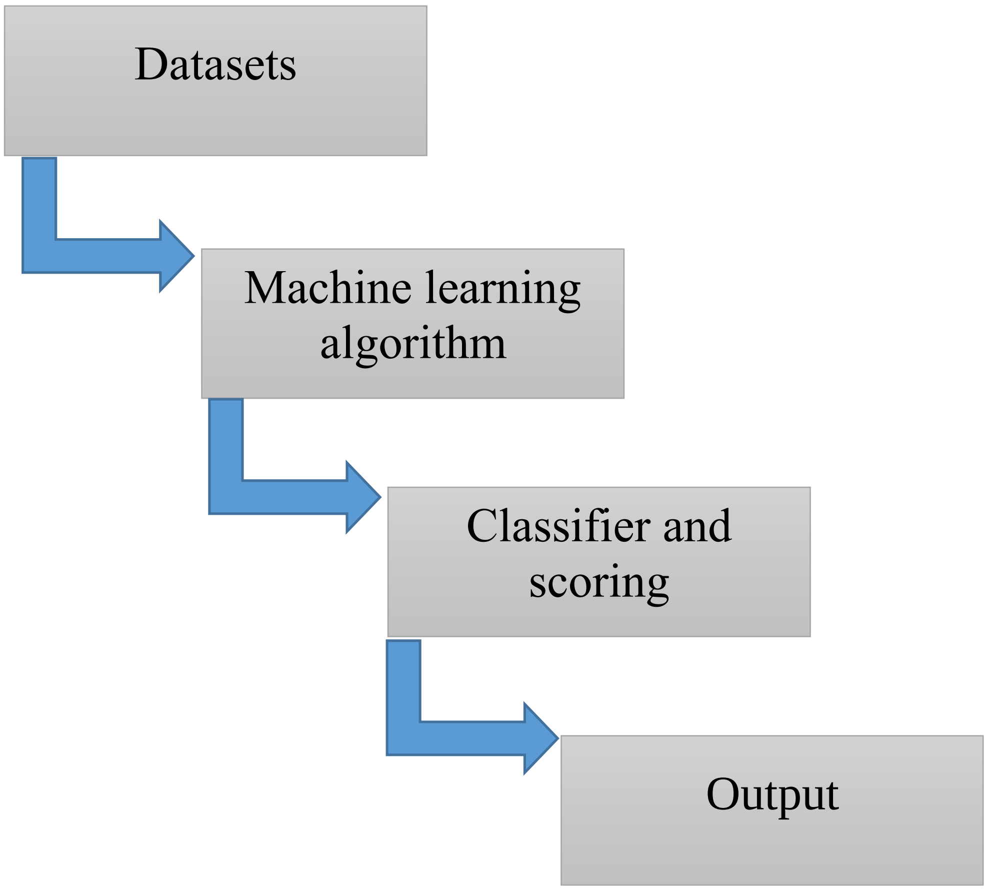


Figure 5.1 system flow

5.2 SETTING UP ENVIRONMENT

The required packages like Natural language toolkit, PyE nchant, Textblob, Six and NLTK data are downloaded. Then the programs are invoked using command prompt.

5.3 SYSTEM IMPLEMENTATION

In the proposed system there is three text files which acts as a datasets and two test datasets. The datasets are then given as a input and the programs that using it are

Main.py -This is the which is responsible for pre-processing the data and produce output. It gives final output as text files which is positive, neutral, negative and feature score.

Hac.py- Here the features are filtered out from reviews and unwanted other words are removed. It contain high adjective count algorithm which does this job and its main role is to produce adjective count.

Mos.py- Here the reviews are the classified and scores for features are calculated. For this maximum opinion score algorithm is used.

Adjscore.py- In this code the meaningless adjectives are removed from the reviews and the score is provided to remaining adjectives.

withNgrams.py- In this phase the scores for the nouns are determined. Nouns are nothing but features of the product.

CHAPTER 6

RESULT AND ANALYSIS

6.1 TESTING THE MODULE

System testing presents an interesting anomaly for the software engineer. The engineer creates a series of test cases that are intended to “demolish” the software that has been built. Testing requires that the developer discards the preconceived notions of the “correctness” of software just developed and overcome a conflict of interest that occurs when errors are uncovered.

There are several rules that can serve well as testing objectives. Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet undiscovered error. A successful test is one that uncovers an as-yet undiscovered error.

Initially the environment checking is done to see if all the necessary requirements are installed or not.

Installation of NLTK, PyEnchat, textblob and numpy are confirmed. The model is then tested. The model created after training is then fed with different text files. The results are obtained and the accuracy is checked. The model is able to identify reviews positivity, negativity and hence the model is tested successfully.

6.2 RESULT ANALYSIS

The system has been tested several times, improvising the result and accuracy at each test. The training accuracy tends to be 80% and the model results coped up with the accuracy.

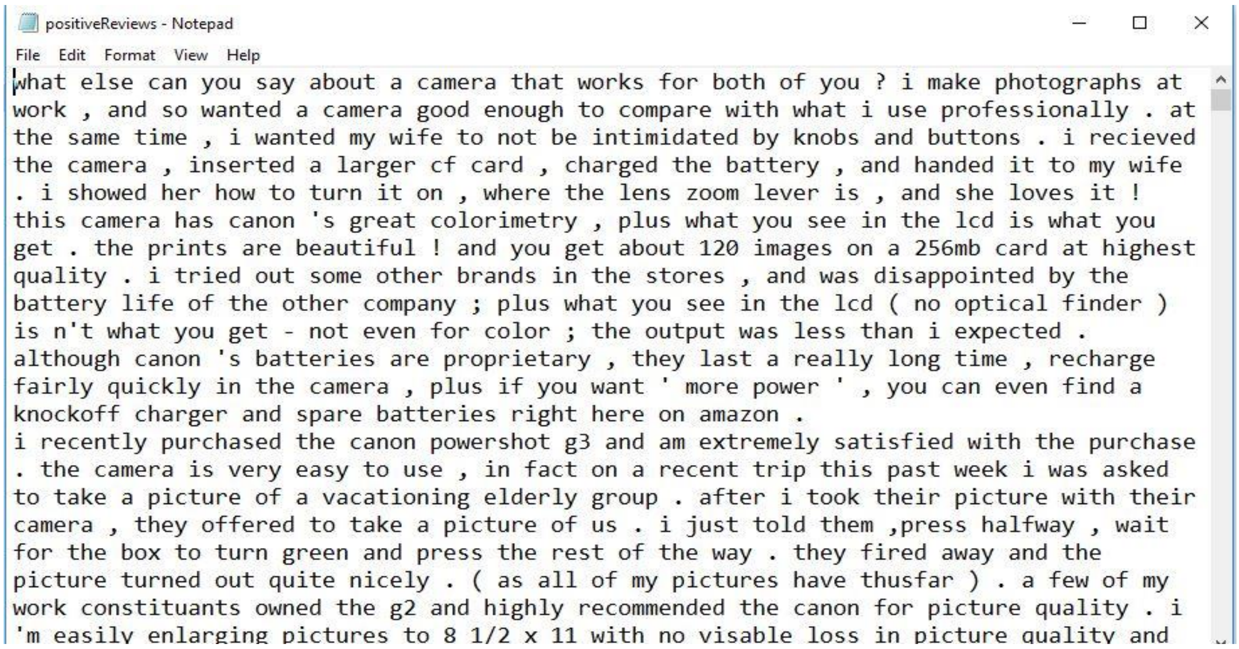
The module classify the reviews into positive , negative and neutral and give feature score.

Now using the reviews about canon camera will going to get the positive ,negative and neutral and feature scores.

Positive reviews:

By recognizing the adjective is positive or negative the positive reviews are found out and get saved in a separate file

The figure 6.1 below shows the positive review report filtered out from mixed reviews.



```
what else can you say about a camera that works for both of you ? i make photographs at
work , and so wanted a camera good enough to compare with what i use professionally . at
the same time , i wanted my wife to not be intimidated by knobs and buttons . i recieved
the camera , inserted a larger cf card , charged the battery , and handed it to my wife
. i showed her how to turn it on , where the lens zoom lever is , and she loves it !
this camera has canon 's great colorimetry , plus what you see in the lcd is what you
get . the prints are beautiful ! and you get about 120 images on a 256mb card at highest
quality . i tried out some other brands in the stores , and was disappointed by the
battery life of the other company ; plus what you see in the lcd ( no optical finder )
is n't what you get - not even for color ; the output was less than i expected .
although canon 's batteries are proprietary , they last a really long time , recharge
fairly quickly in the camera , plus if you want ' more power ' , you can even find a
knockoff charger and spare batteries right here on amazon .
i recently purchased the canon powershot g3 and am extremely satisfied with the purchase
. the camera is very easy to use , in fact on a recent trip this past week i was asked
to take a picture of a vacationing elderly group . after i took their picture with their
camera , they offered to take a picture of us . i just told them ,press halfway , wait
for the box to turn green and press the rest of the way . they fired away and the
picture turned out quite nicely . ( as all of my pictures have thusfar ) . a few of my
work constituants owned the g2 and highly recommended the canon for picture quality . i
'm easily enlarging pictures to 8 1/2 x 11 with no visable loss in picture quality and
```

Figure 6.1 positive review

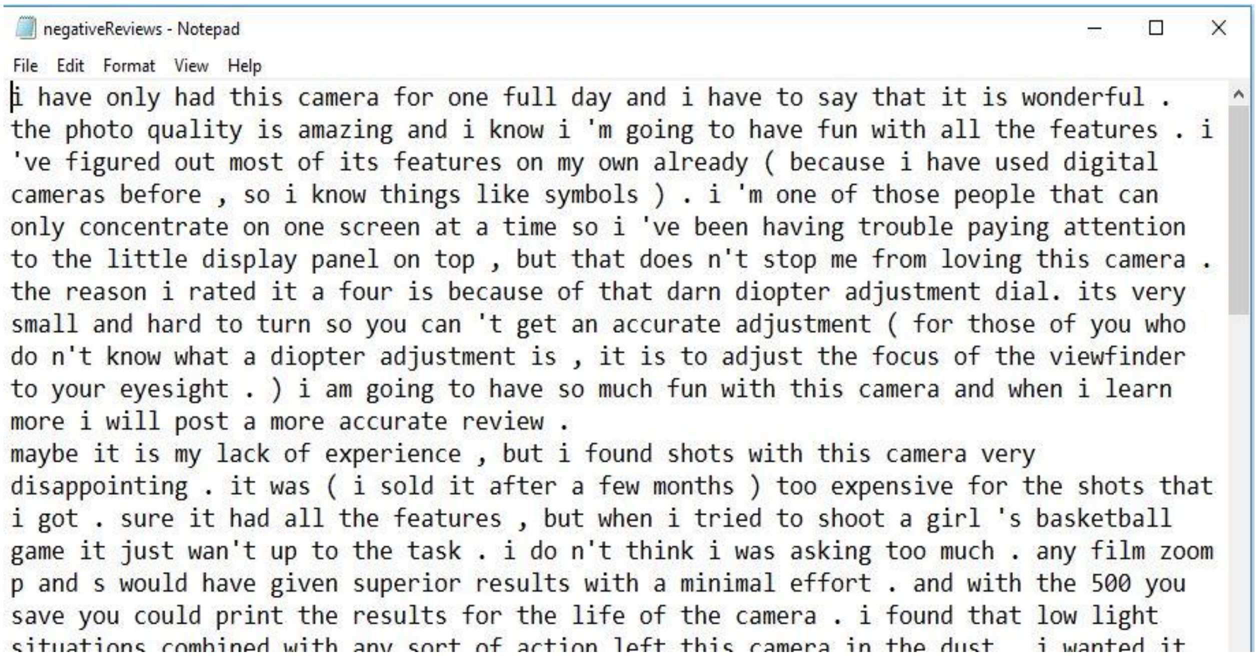
Negative reviews:

When the module found inversion words in the review it recognize it as an negative review. Inversion words are the words which are associated with negations like not,no etc,.

It also find negative review by using adjectives which implies negative meaning.

The inversion words can be used to find whether the review is negative and also again it is an adjective too. So if more inversion words are related to noun or feature the scores will get reduce.

The below figure 6.2 shows the negative review filtered out.



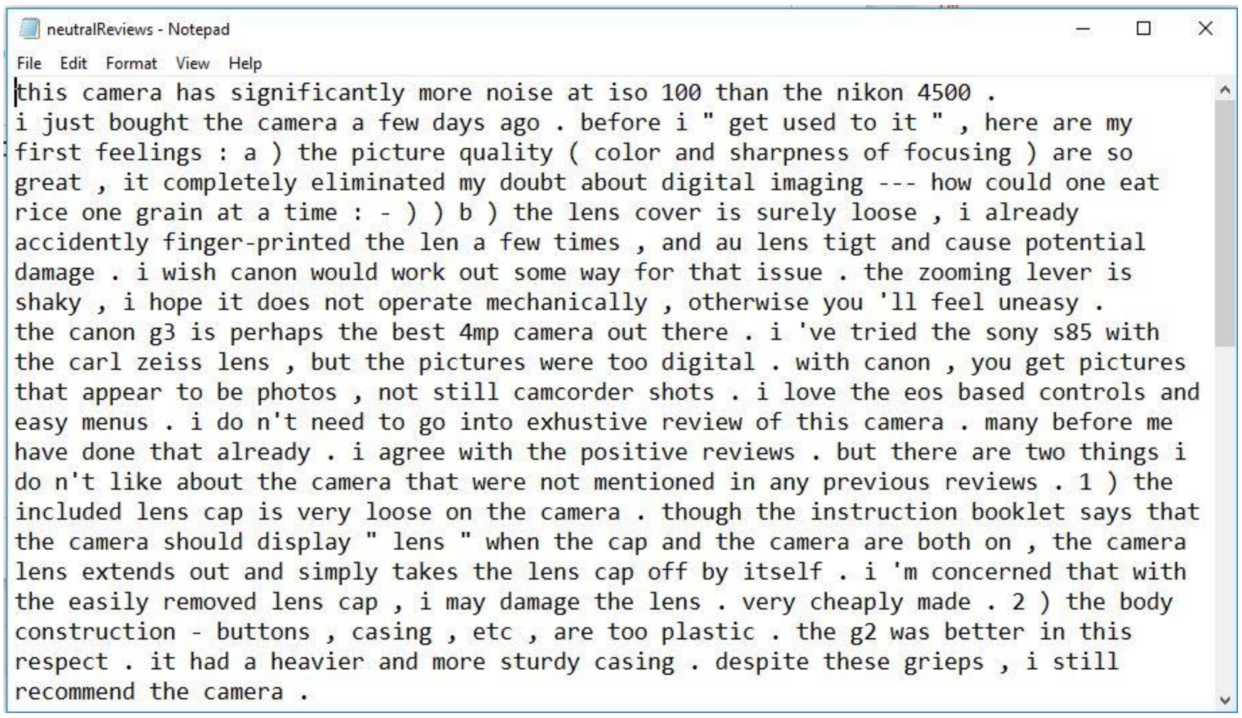
```
i have only had this camera for one full day and i have to say that it is wonderful .
the photo quality is amazing and i know i 'm going to have fun with all the features . i
've figured out most of its features on my own already ( because i have used digital
cameras before , so i know things like symbols ) . i 'm one of those people that can
only concentrate on one screen at a time so i 've been having trouble paying attention
to the little display panel on top , but that does n't stop me from loving this camera .
the reason i rated it a four is because of that darn diopter adjustment dial. its very
small and hard to turn so you can 't get an accurate adjustment ( for those of you who
do n't know what a diopter adjustment is , it is to adjust the focus of the viewfinder
to your eyesight . ) i am going to have so much fun with this camera and when i learn
more i will post a more accurate review .
maybe it is my lack of experience , but i found shots with this camera very
disappointing . it was ( i sold it after a few months ) too expensive for the shots that
i got . sure it had all the features , but when i tried to shoot a girl 's basketball
game it just wan't up to the task . i do n't think i was asking too much . any film zoom
p and s would have given superior results with a minimal effort . and with the 500 you
save you could print the results for the life of the camera . i found that low light
situations combined with any sort of action left this camera in the dust . i wanted it
```

Figure 6.2 Negative review

Neutral reviews:

The neutral review is does not positive and at the same time it is not negative.

The figure 6.3 below shows the neutral review output.



```
this camera has significantly more noise at iso 100 than the nikon 4500 .
i just bought the camera a few days ago . before i " get used to it " , here are my
first feelings : a ) the picture quality ( color and sharpness of focusing ) are so
great , it completely eliminated my doubt about digital imaging --- how could one eat
rice one grain at a time : - ) ) b ) the lens cover is surely loose , i already
accidently finger-printed the len a few times , and au lens tigt and cause potential
damage . i wish canon would work out some way for that issue . the zooming lever is
shaky , i hope it does not operate mechanically , otherwise you 'll feel uneasy .
the canon g3 is perhaps the best 4mp camera out there . i 've tried the sony s85 with
the carl zeiss lens , but the pictures were too digital . with canon , you get pictures
that appear to be photos , not still camcorder shots . i love the eos based controls and
easy menus . i do n't need to go into exhustive review of this camera . many before me
have done that already . i agree with the positive reviews . but there are two things i
do n't like about the camera that were not mentioned in any previous reviews . 1 ) the
included lens cap is very loose on the camera . though the instruction booklet says that
the camera should display " lens " when the cap and the camera are both on , the camera
lens extends out and simply takes the lens cap off by itself . i 'm concerned that with
the easily removed lens cap , i may damage the lens . very cheaply made . 2 ) the body
construction - buttons , casing , etc , are too plastic . the g2 was better in this
respect . it had a heavier and more sturdy casing . despite these grieps , i still
recommend the camera .
```

Figure 6.3 Neutral review

Feature score:

Finally the feature score plays the major role in giving short and clear idea about the product. The score ranges from maximum score 4 to minimum score - 4

S.no	Feature	Score
1	Life	3.600
2	Feel	3.200
3	Zoom	2.920
4	Auto	2.600
5	Canon	2.560
6	View finder	2.200
7	Month	2
8	Resolution	1.920
9	Quality	1.751
10	Use	1.695
11	Plenty	1.600
12	Camera	1.530

Table 6.2.4 Feature score positive

In the above table 6.2.4 the features are getting positive scores and the feature life of the camera gets maximum score.

S.no	**Feature**	**Score**
1	picture	-0.375
2	Lcd	-0.300
3	battery	-0
4	Dial	-0.042
5	mode	-0.100
6	Strap	-0.200
7	Card	-0.267
8	focus	-0.267
9	charge	-0.286
10	Bit	-1.200
11	Line	-1.333
12	hand	-2.800
13	Lot	-3.200

Table 6.2.5 feature score negative

In the above table 6.2.5 some features of camera are getting negative scores which implies these features are not good in that camera.

S.no	**Feature**	**Score**
1	review	1.195
2	Time	1.131
3	priority	1.120
4	screen	1.100
5	button	1
6	point	0.933
7	software	0.871
8	thing	0.850
9	control	0.800
10	flash	0.525
11	picture	0.375
12	Lcd	0.300
13	battery	0

Table 6.2.6 Feature score neutral

In the above table 6.2.6 the features are getting neutral scores.

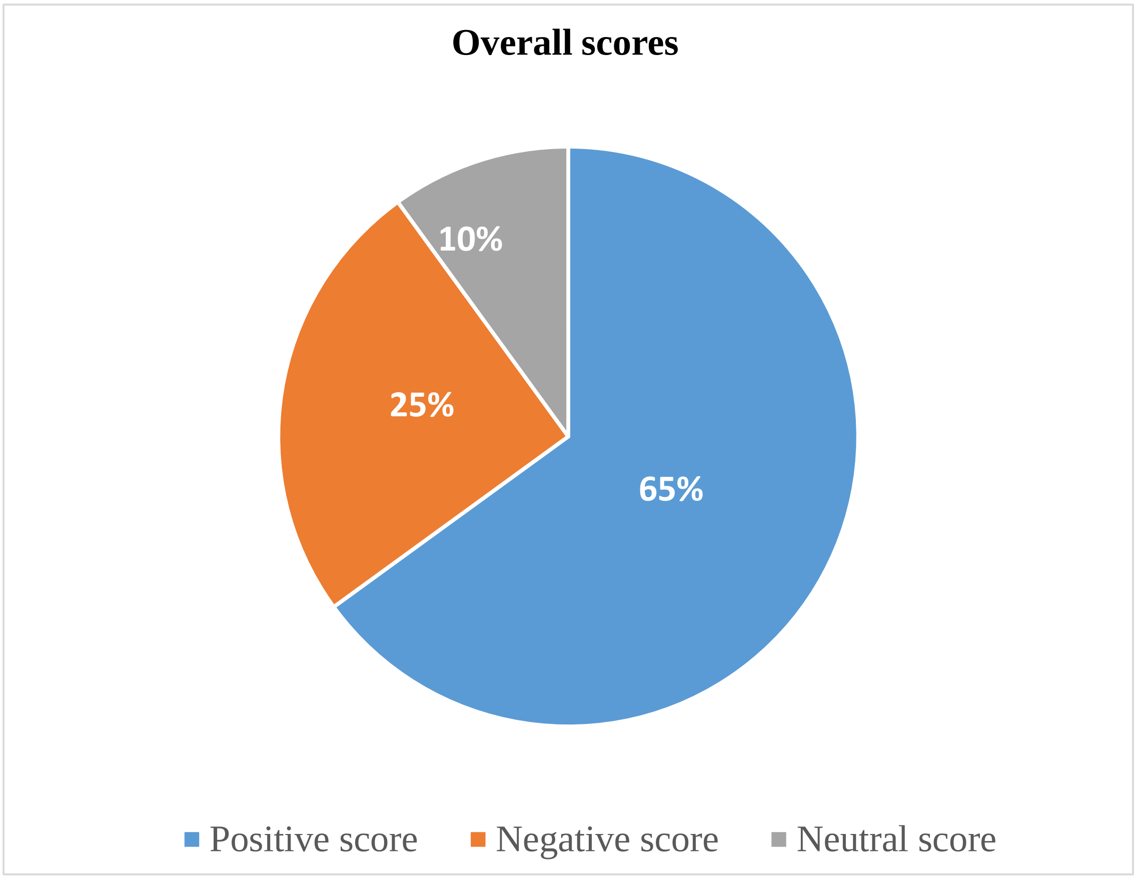


Figure 6.4 Score chart

The above pie chart shows overall scores the camera got and percentage of scores. In this chart the percentage of positive score is higher than other type of scores. So the quality of camera is good and customer can bought this without any doubt.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The system is a state-of-the-art review analysis system that is much better than the evaluated review systems of some E-commerce giants. On the basis of different features being selected, will be able to determine the accurate performance and customer satisfaction of the product. This would help to establish a proper review system for genuine customers, who are worried about the product performance. Also, this would be making it possible for different E-commerce brands to establish them and the products they sell. This system would enable users to make a faster, more accurate, and by far a more desirable choice. It would also eliminate agenda driven campaigns, and would move the online shopping platform towards an honest and more trust-based direction.

7.2 FUTURE ENHANCEMENT

In future, the work can be extended to perform multi-class classification of reviews which will provide delineated nature of review to the consumer, hence leads to better judgement of the product. It can also be used to predict rating of a product from the review. This will provide users with reliable rating because sometimes the rating received by the product and the sentiment of the review do not provide justice to each other. The proposed extension of work will be very beneficial for the e-commerce industry as it will augment user satisfaction and trust.

APPENDIX

Main.py

```
import os
import sys
import HAC
import math
import FileCreationWithBigrams
import AdjScore
import operator
import collections
from textblob import TextBlob
from texttable import Texttable
filename = sys.argv[1]
print(filename)
reviewTitle = []
reviewContent = []
posCount = 0
negCount = 0
neutCount = 0
curIndex = -1
posActIndex = []
negActIndex = []
neutActIndex = []

with open(filename) as f:
```

```
review = []
for line in f:
    if line[:6] == "[+][t]":

        if review:
            reviewContent.append(review)
            review = []
        reviewTitle.append(line.split("[+][t]")[1].rstrip("\r\n"))
        posCount += 1
        curIndex += 1
        posActIndex.append(curIndex)
    elif line[:6] == "[-][t]":
        if review:
            reviewContent.append(review)
            review = []
        reviewTitle.append(line.split("[-][t]")[1].rstrip("\r\n"))
        negCount += 1
        curIndex += 1
        negActIndex.append(curIndex)
    elif line[:6] == "[N][t]":
        if review:
            reviewContent.append(review)
            review = []
        reviewTitle.append(line.split("[N][t]")[1].rstrip("\r\n"))
        neutCount += 1
        curIndex += 1
        neutActIndex.append(curIndex)
```

```
            else:
                if "##" in line:

                    x = line.split("##")
                    for i in range(1, len(x)):
                        review.append(x[i].rstrip("\r\n"))
                else:
                    continue
        reviewContent.append(review)
FileCreationWithBigrams.fileCreation(reviewContent,filename)
import MOS
import WithNgrams
adjDict = HAC.findFeatures(reviewContent,filename)
featureList = WithNgrams.getList()
adjScores = AdjScore.getScore(adjDict,filename)
posPredIndex, negPredIndex, neutPredIndex, avgFeatScore =
MOS.rankFeatures(adjScores, featureList, reviewTitle,
reviewContent)outputDir = "./Results_" + filename
if not os.path.exists(outputDir):
   os.makedirs(outputDir)
with open(outputDir + "/positiveReviews.txt", "w") as filePos:
        for i in posPredIndex:
            for k in range(len(reviewContent[i])):
                filePos.write(reviewContent[i][k])
            filePos.write("\n")
with open(outputDir + "/negativeReviews.txt", "w") as fileNeg:
        for i in negPredIndex:
            for k in range(len(reviewContent[i])):
                fileNeg.write(reviewContent[i][k])
            fileNeg.write("\n")
with open(outputDir + "/neutralReviews.txt", "w") as fileNeut:
        for i in neutPredIndex:
            for k in range(len(reviewContent[i])):
                fileNeut.write(reviewContent[i][k])
            fileNeut.write("\n")
with open(outputDir + "/featureScore.txt", "w") as fileFeat:
        t = Texttable()
        lst = [["Feature", "Score"]]
        for tup in avgFeatScore:
            lst.append([tup[0], tup[1]])
        t.add_rows(lst)
        fileFeat.write(str(t.draw()))
print("The files are successfully created in the dir '" + outputDir + "'")
```

Adjscore.py

```
from textblob import TextBlob
import operator
from collections import OrderedDict
#Return the adjective scores for adjectives in adjList
def getScore(adjList,filename):
    inversion_words = []
    adjScores = dict()
    for i in adjList:
        blob = TextBlob(i)
```

```
            if(blob.sentiment.polarity != 0):
                #If the adjective expresses some polarity
                adjScores[i] = blob.sentiment.polarity

        #Multiply the polarity by 4 to get in the given range
        adjScores.update((x, 4 * y) for x, y in adjScores.items())
        if filename == "CanonG3.txt" or filename == "Nikon.txt":

            adjScores  = OrderedDict([('awesome', 4.0), ('excellent', 4.0),
('wonderful', 4.0), ('ideal', 3.6), ('incredible', 3.6),
                ('beautiful', 3.4), ('great', 3.2), ('happy', 3.2), ('bright',
2.8000000000000003), ('nice', 2.4),
                ('love', 2.0), ('creative', 2.0), ('able', 2.0), ('satisfied',
2.0), ('pleased', 2.0), ('sophisticated', 2.0),
                ('easy', 1.7333333333333334), ('fantastic', 1.6),
('advanced', 1.6), ('fabulous', 1.6), ('light', 1.6),
                ('comfortable', 1.6), ('available', 1.6), ('clean',
1.4666666666666668), ('quick', 1.3333333333333333),
                ('super', 1.3333333333333333), ('worth', 1.2),
('powerful', 1.2), ('first', 1.0), ('positive', 0.9090909090909091),
                ('ready', 0.8), ('real', 0.8), ('reasonable', 0.8), ('fast',
0.8), ('much', 0.8), ('main', 0.6666666666666666),
                ('high', 0.64), ('live', 0.5454545454545454), ('clear',
0.4000000000000001), ('flat', -0.1), ('minor', -0.2),
                ('single', -0.2857142857142857), ('remote', -0.4),
('partially', -0.4), ('extreme', -0.5), ('sharp', -0.5),
                ('average', -0.6), ('heavy', -0.8),('flaw', -0.8), ('harsh', -
0.8), ('raw', -0.9230769230769231), ('small', -1.0),
                ('hard', -1.1666666666666667), ('slow', -
1.2000000000000002), ('serious', -1.3333333333333333),
                ('disappointing', -2.4), ('bad', -2.7999999999999994)])
```

```
	adjScores = sorted(adjScores.items(), key=operator.itemgetter(1), reverse=True)


	#Print the adjectives and their scores

	return(OrderedDict(adjScores))
```

HAC.py

```
import re

import nltk

import string

import enchant

import operator

from nltk.corpus import stopwords

from collections import OrderedDict

from textblob import TextBlob, Word

from nltk.corpus import brown

from textblob import Blobber

from textblob.taggers import NLTKTagger


#Dict to convert the raw user text to meaningful words for analysis

apostropheList = {"n't" : "not","aren't" : "are not","can't" : "cannot","couldn't" : "could not","didn't" : "did not","doesn't" : "does not", \"don't" : "do not","hadn't" : "had not","hasn't" : "has not","haven't" : "have not","he'd" : "he had","he'll" : "he will", \"he's" : "he is","I'd" : "I had","I'll" : "I will","I'm" : "I am","I've" : "I have","isn't" : "is not","it's" : \"it is","let's" : "let us","mustn't" : "must not","shan't" : "shall not","she'd" : "she had","she'll" : "she will", \"she's" : "she is", "shouldn't" : "should not","that's" : "that is","there's" : "there is","they'd" : "they had", \"they'll" : "they will", "they're" : "they are","they've" : "they
have","we'd" : "we had","we're" : "we are","we've" : "we have",
\"weren't" : "were not", "what'll" : "what will","what're" : "what
are","what's" : "what is","what've" : "what have", \"where's" : "where
is","who'd" : "who had", "who'll" : "who will","who're" : "who
are","who's" : "who is","who've" : "who have", \ "won't" : "will
not","wouldn't" : "would not", "you'd" : "you had","you'll" : "you
will","you're" : "you are","you've" : "you have"}


#Removing stop words might lead to better data analysis

stopWords = stopwords.words("english")

#Exclude punctuations from the reviews

exclude = set(string.punctuation)

exclude.remove("_")

#Remove all the hyperlinks from the reviews

linkPtrn = re.compile("^(https?:\/\/)?([\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-
]*)*\/?$")

#English vocabulary

enchVocab = enchant.Dict("en_US")

vocabList = set(w.lower() for w in nltk.corpus.words.words())

#Max hops to find the nearby noun from the position of adjective

maxHops = 4

#Find all the potential features from the reviews

def findFeatures(reviewContent,filename):

        #nounScores is the dict containing nouns from all reviews and their
respective scores from HAC algorithm

        nounScores = dict()

        #adjDict dict contains adjective and the corresponding noun which
it is assigned to

        adjDict = dict()

        tb = Blobber(pos_tagger = NLTKTagger())
        for a in range(len(reviewContent)):
                #Stores the score of the nouns

            for i in range(len(reviewContent[a])):

                text = ' '.join([word for word in
    reviewContent[a][i].split() if word not in stopwords.words("english")])

                text = ''.join(ch for ch in text if ch not in exclude)

                text = nltk.word_tokenize(text)

                x = nltk.pos_tag(text)

                #Get the noun/adjective words and store it in tagList

                tagList = []

                for e in x:

                    if(e[1] == "NN" or e[1] == "JJ"):

                        tagList.append(e)

         #Add the nouns(which are not in the nounScores dict) to the dict

                for e in tagList:

                    if e[1] == "NN":

                        if e[0] not in nounScores:

                            nounScores[e[0]] = 0

                 #For every adjective, find nearby noun

                for l in range(len(tagList)):

                    if("JJ" in tagList[l][1]):

                        j = k = leftHop = rightHop = -1

         #Find the closest noun to the right of the adjective in the line

                        for j in range(l + 1, len(tagList)):

                            if(j == l + maxHops):

                                break
```

```
                                                  if("NN" in tagList[j][1]):
                                                          rightHop = (j - l)
                                                          Break
#Find the closest noun to the left of the adjective in the line
                                          for k in range(l - 1, -1, -1):
#Incase hopped the 'maxHops' number of words and no noun was found, ignore the adjective
                                                  if(j == l - maxHops):
                                                          break
                                                  if("NN" in tagList[k][1]):
                                                          leftHop = (l - k)
                                                          break
#Compare which noun is closer to adjective(left or right) and assign the adj to corresponding noun
                                          if(leftHop > 0 and rightHop > 0):
                                  #If nouns exist on both sides of adjective
                                              if (leftHop - rightHop) >= 0:
                                  #If left noun is farther
                                                 adjDict[tagList[l][0]] = tagList[j][0]
                                                  nounScores[tagList[j][0]] += 1
                                              else:
                                                          #If right noun is farther
                                                 adjDict[tagList[l][0]]= tagList[k][0]
                                                  nounScores[tagList[k][0]] += 1
                                          elif leftHop > 0:
                                  #If noun is not found on RHS of adjective
                                                 adjDict[tagList[l][0]]= tagList[k][0]
                                                  nounScores[tagList[k][0]] += 1
```

```
                                        elif rightHop > 0:
                                        #If noun is not found on LHS of adjective

                                                adjDict[tagList[l][0]] = tagList[j][0]

                                                 nounScores[tagList[j][0]] += 1


        nounScores=OrderedDict(sorted(nounScores.items(),key=operator.
itemgetter(1)))

        return filterAdj(nounScores, adjDict,filename)

def filterAdj(nounScores, adjDict,filename):

        adjectList = list(adjDict.keys())

        nouns = []

        for key, value in nounScores.items():

                if value >= 3:

                        nouns.append(key)

        nouns1 = ["sound quality","battery life","great phone","cell
phone","menu option","color screen","flip phone","samsung
phone","nokia phones","corporate email","ring tone","tmobile service"]

        nouns = set(nouns)

        stopWords = stopwords.words("english")

        exclude = set(string.punctuation)

        reviewTitle = []

        reviewContent = []

        with open(filename) as f:

                review = []

                for line in f:

                        if line[:6] == "[+][t]":

                                if review:

                                        reviewContent.append(review)
                        review = []

    reviewTitle.append(line.split("[+][t]")[1].rstrip("\r\n"))
                elif line[:6] == "[-][t]":
                    if review:
                        reviewContent.append(review)
                        review = []
                    reviewTitle.append(line.split("[-
][t]")[1].rstrip("\r\n"))
                else:
                    if "##" in line:
                        x = line.split("##")
                        #if len(x[0]) != 0:
                        for i in range(1, len(x)):
                            review.append(x[i].rstrip("\r\n"))
                    else:
                        continue
            reviewContent.append(review)

    tb = Blobber(pos_tagger=NLTKTagger())
    nounScores = dict()
    f = open('modified.txt', 'w')
    for a in range(len(reviewContent)):
        f.write("[t]"+reviewTitle[a])
        f.write("\r\n")
        #Stores the score of the nouns
        for i in range(len(reviewContent[a])):
```

```
                    text = reviewContent[a][i]
                    x = tb(text).tags #Perceptron tagger
                    #Get the noun/adjective words and store it in tagList
                    tagList = []
                    e = 0
                    f.write("##")
                    while e<len(x):
                        tagList = []
                        f.write(x[e][0])
                        e = e+1
                        count = e
                        if(count<len(x) and x[count-1][1] == "NN" and
x[count][1] == "NN"):
                          tagList.append(x[count-1][0])

                         while(count < len(x) and x[count][1]== "NN"):
                                tagList.append(x[count][0])
                                count = count+1
                        if tagList != [] and len(tagList) == 2:
                            if set(tagList) <= nouns:
                                for t in range(1,len(tagList)):
                                    f.write(tagList[t])
                                e = count
                        f.write(" ")
                    f.write(".\r\n")
    return adjectList
```

MOS.py

```
import re

import string

import operator

from collections import OrderedDict

from textblob import TextBlob

from nltk.corpus import stopwords

import os




#Dict to convert the raw user text to meaningful words for analysis

apostropheList = {"n't" : "not","aren't" : "are not","can't" :
"cannot","couldn't" : "could not","didn't" : "did not","doesn't" : "does
not", \"don't" : "do not","hadn't" : "had not","hasn't" : "has not","haven't" :
"have not","he'd" : "he had","he'll" : "he will", \ "he's" : "he is","I'd" : "I
had","I'll" : "I will","I'm" : "I am","I've" : "I have","isn't" : "is not","it's" :
\ "it is","let's" : "let us","mustn't" : "must not","shan't" : "shall
not","she'd" : "she had","she'll" : "she will", \"she's" : "she is", "shouldn't"
: "should not","that's" : "that is","there's" : "there is","they'd" : "they had",
\ "they'll" : "they will", "they're" : "they are","they've" : "they
have","we'd" : "we had","we're" : "we are","we've" : "we have",
\"weren't" : "were not", "what'll" : "what will","what're" : "what
are","what's" : "what is","what've" : "what have", \ "where's" : "where
is","who'd" : "who had", "who'll" : "who will","who're" : "who
are","who's" : "who is","who've" : "who have", \"won't" : "will
not","wouldn't" : "would not", "you'd" : "you had","you'll" : "you
will","you're" : "you are","you've" : "you have"}

#Removing stop words might lead to better data analysis

stopWords = stopwords.words("english")

#Exclude punctuations from the reviews

exclude = set(string.punctuation)

#The two lists which holds the review title and the corresponding reviews

reviewTitle = []

reviewContent = []

alpha = 0.6

with open("modified.txt") as f:

        review = []

        for line in f:

                if line[:3] == "[t]":

                        if review:

                                reviewContent.append(review)

                                review = []

                        reviewTitle.append(line.split("[t]")[1].rstrip("\r\n"))

                else:
```

```
            if "##" in line:
                x = line.split("##")
                for i in range(1, len(x)):
                    review.append(x[i].rstrip("\r\n"))
            else:
                continue
    reviewContent.append(review)
def rankFeatures(adj_scores, features, reviewTitle, reviewContent):
    #Lists containing indices of the reviewContent list
    pos_review_index = dict()
    neg_review_index = dict()
    neut_review_index = dict()
    #scores for a feature from all the reviews
    global_noun_scores = dict()
    #Number of adj describing a feature obtained from all the reviews
    global_noun_adj_count = dict()
    #Iterate for each review in the list of reviews
    for a in range(len(reviewContent)):
        #scores for a feature from the review
        review_noun_scores = dict()
        title_noun_scores = dict()

        #Number of adj describing a feature in the review

        review_noun_adj_count = dict()

        title_noun_adj_count = dict()

        #Iterate for all lines in a review

        for lineIndex in range(len(reviewContent[a]) + 1):

            if(lineIndex == len(reviewContent[a])):

                line_words = reviewTitle[a]

            else:

                line_words = reviewContent[a][lineIndex]

            line_words = ' '.join([apostropheList[word] if word in
apostropheList else word for word in line_words.split()])

            line_words = ''.join(ch for ch in line_words if ch not in
exclude)

            line_words = re.sub(r' [a-z][$]? ', ' ', line_words)

            line_words = [word for word in line_words.split()
if(word not in stopwords.words("english") and not word.isdigit()) and
len(word) > 2]

            #Iterate for each word in the line

            for wordIndex in range(len(line_words)):

                word = line_words[wordIndex]
```

```
#If the word is an adj, get the adj score and check for inversion words
onto its left
                                if word in adj_scores:
                                        score = adj_scores[word]
                                 #Left context for inversion words
                                        if(wordIndex - 2 >= 0):
                               #Phrase is the last two words and the present adj
                                                phrase = line_words[wordIndex -
2] + " " + line_words[wordIndex - 1] + " " + line_words[wordIndex]
                      #If the polarity of the phrase and the adj is opposite
        if((TextBlob(phrase).sentiment.polarity * score) < 0):
                                                        score *= -1
                                        elif(wordIndex - 1 >= 0):


                                #Phrase is the last word and the present adj
                                                phrase = line_words[wordIndex -
1] + " " + line_words[wordIndex]
#If the polarity of the phrase and the adj is opposite
                                        if((TextBlob(phrase).sentiment.polarity *
score) < 0):
                                                        score *= -1
                                        #Find the closest feature to the adj

                                        closest_noun =
find_closest_noun(wordIndex, line_words, features)

                                        if(closest_noun is None):

                                             Continue

                                        if(lineIndex == len(reviewContent[a])):

                                             if(closest_noun in
title_noun_scores):

     title_noun_scores[closest_noun] += score

                                             else:

     title_noun_scores[closest_noun] = score

#Increase the count of no of adjs describing the feature

                                             if(closest_noun in
title_noun_adj_count):


     title_noun_adj_count[closest_noun] += 1

                                             else:

     title_noun_adj_count[closest_noun] = 1

                                             else:

#Update the score of the feature which the adj is describing

                                             if(closest_noun in
review_noun_scores):
        review_noun_scores[closest_noun] += score
                                             else:
        review_noun_scores[closest_noun] = score
                                             if(closest_noun in
global_noun_scores):

        global_noun_scores[closest_noun] += score
                                             else:
        global_noun_scores[closest_noun] = score
#Increase the count of no of adjs describing the feature
                                             if(closest_noun in
review_noun_adj_count):
        review_noun_adj_count[closest_noun] += 1
                                             else:
        review_noun_adj_count[closest_noun] = 1
                                             if(closest_noun in
global_noun_adj_count):
        global_noun_adj_count[closest_noun] += 1
                                             else:
        global_noun_adj_count[closest_noun] = 1
```

```
                #Score for the review content
                total_score = sum(review_noun_scores.values())
                total_adj = sum(review_noun_adj_count.values())
                if(total_adj == 0):
                        review_score = 0
                else:
                        review_score = total_score / float(total_adj)
        #Find the title score
                title_total_score = sum(title_noun_scores.values())
                title_total_adj = sum(title_noun_adj_count.values())
                 if(title_total_adj == 0):
                        title_score = 0
                else:
                        title_score = title_total_score / float(title_total_adj)
          #The total score for the review
                avg_score = ((alpha * title_score) + review_score) / (alpha +
1)

        #Incase both title_score and review_scores are 0's, then ignore that
review
                if(avg_score == 0):
                        neut_review_index[a] = avg_score

                        continue

                 if(avg_score > 0):

                        pos_review_index[a] = avg_score

                else:

                        neg_review_index[a] = avg_score

#Scores for each feature from all the reviews

        avg_feature_score = dict()

        for noun in global_noun_scores:

                avg_feature_score[noun] = global_noun_scores[noun] /
float(global_noun_adj_count[noun])

        avg_feature_score = sorted(avg_feature_score.items(),
key=operator.itemgetter(1), reverse=True)

pos_review_index = OrderedDict(sorted(pos_review_index.items(),
key=operator.itemgetter(1), reverse=True))

        neg_review_index =
OrderedDict(sorted(neg_review_index.items(),
key=operator.itemgetter(1)))

        posPredIndex = []

        negPredIndex = []

        neutPredIndex = []

#Gather the review index only (not score) from dict
```

```
        for i, j in pos_review_index.items():
                posPredIndex.append(i)
        for i, j in neg_review_index.items():
                negPredIndex.append(i)
        for i, j in neut_review_index.items():
                neutPredIndex.append(i)
#Remove the temp file
        os.remove("modified.txt")
        return posPredIndex, negPredIndex, neutPredIndex,
avg_feature_score
#Find the closest feature for an adj. Assumes a noun is found within 3
steps from the adj.
def find_closest_noun(wordIndex, line_words, features):
        ptr = 1
        while(ptr <= 3):
                if(wordIndex + ptr < len(line_words) and
line_words[wordIndex + ptr] in features):
                        return line_words[wordIndex + ptr]
                elif(wordIndex - ptr >= 0 and line_words[wordIndex - ptr] in
features):
                        return line_words[wordIndex - ptr]
                else:
                        ptr += 1
```

WithNgrams.py

```
import nltk

from nltk.corpus import stopwords

import string

import operator

from collections import OrderedDict

#from textblob import TextBlob

from textblob import Blobber

from textblob.taggers import PatternTagger, NLTKTagger

stopWords = stopwords.words("english")

exclude = set(string.punctuation)

reviewTitle = []

reviewContent = []

def getList():

        #reading from the created file "modified.txt"

        with open("modified.txt") as f:

                review = []

                for line in f:

                        if line[:3] == "[t]":
```

```
                    if review:
                        reviewContent.append(review)
                        review = []
        reviewTitle.append(line.split("[t]")[1].rstrip("\r\n"))
                else:
                    if "##" in line:
                        x = line.split("##")
                        for i in range(1, len(x)):
                            review.append(x[i].rstrip("\r\n"))
                    else:
                        continue
            reviewContent.append(review)
        tb = Blobber(pos_tagger=NLTKTagger())
        nounScores = dict()
        for a in range(len(reviewContent)):
                #Stores the score of the nouns
            for i in range(len(reviewContent[a])):
                #text = reviewContent[a][i]
                text = ' '.join([word for word in
reviewContent[a][i].split() if word not in stopwords.words("english")])
                text = ''.join(ch for ch in text if ch not in exclude)
```

```
                    text = nltk.word_tokenize(text)

                    x = nltk.pos_tag(text)

                    #x = TextBlob(text).tags #textblob tagger

                    #x = tb(text).tags #Perceptron tagger

                    #Get the noun/adjective words and store it in tagList

                    tagList = []

                    for e in x:

                        if(e[1] == "NN" or e[1] == "JJ"):

                            tagList.append(e)

            #Add the nouns(which are not in the nounScores dict) to the dict

                    for e in tagList:

                        if e[1] == "NN":

                            if e[0] not in nounScores:

                                nounScores[e[0]] = 0

        #For every adjective, find nearby noun

                    l=0

                    for l in range(len(tagList)):

                        if(tagList[l][1] == "JJ"):

                            check=0

                            j = 0
```

```
k = 0

ct1 = 0

for j in range(l + 1, len(tagList)):

    if ct1 == 4:

        break

    if(tagList[j][1] == "NN"):

    #nounScores[tagList[j][0]] += 1

        check = 1

        break

ct = 0

if(l > 0):

    if j == 0:

        j = len(tagList)

    for k in range(l - 1, 0, -1):

        if ct == 4:

            break

        ct += 1

        if(tagList[k][1] == "NN"):

            if(j != len(tagList)):


nounScores[tagList[min(j, k)][0]] += 1
```

```
                                        else:

    nounScores[tagList[k][0]] += 1
                                        break
                            elif check == 1:
                                nounScores[tagList[j][0]] += 1
    nounScores = OrderedDict(sorted(nounScores.items(),
key=operator.itemgetter(1)))
    nouns = []
    for key, value in nounScores.items():
        if value >= 3:
            nouns.append(key)
    return nouns
def intersect(a, b):
  return list(set(a) & set(b))
```

REFERENCES

[1] K. L. S. Kumar, J. Desai and J. Majumdar, "Opinion mining and sentiment analysis on online customer review," 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Chennai, 2016, pp. 1-4.

[2] Z. Singla, S. Randhawa and S. Jain, "Statistical and sentiment analysis of consumer product reviews," 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Delhi, 2017, pp. 1-6.

[3] P. P. Surya and B. Subbulakshmi, "Sentimental Analysis using Naive Bayes Classifier," 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 2019, pp. 1-5.

[3] Abinash Tripathya, Ankit Agrawalb,Santanu Kumar Rath “ Classification of Sentimental Reviews Using Machine Learning Techniques ”, 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015).

[4] M. S. Neethu and R. Rajasree, "Sentiment analysis in twitter using

machine learning techniques," 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, 2013, pp. 1-5.

[5] A. Angelpreethi and S. B. R. Kumar, "An Enhanced Architecture for Feature Based Opinion Mining from Product Reviews," 2017 World Congress on Computing and Communication Technologies (WCCCT), Tiruchirappalli, 2017, pp. 89-92.

[6] Y. Li, X. Feng and S. Zhang, "Detecting Fake Reviews Utilizing Semantic and Emotion Model," 2016 3rd International Conference on Information Science and Control Engineering (ICISCE), Beijing, 2016, pp. 317-320.

[7] Wahyuni, Eka & Djunaidy, Arif. (2016). Fake Review Detection from a Product Review Using Modified Method of Iterative Computation Framework. MATEC Web of Conferences. 58. 03003. 10.1051/matecconf/20165803003.

[8] A. Agarwal, V. Sharma, G. Sikka and R. Dhir, "Opinion mining of news headlines using SentiWordNet," 2016 Symposium on Colossal Data Analysis and Networking (CDAN), Indore, 2016, pp. 1-5.

[9] P. Kherwa, A. Sachdeva, D. Mahajan, N. Pande and P. K. Singh, "An approach towards comprehensive sentimental data analysis and

opinion mining," 2014 IEEE International Advance Computing Conference (IACC), Gurgaon, 2014, pp. 606-612.

[10] Kim S-M, Hovy E (2004) Determining the sentiment of opinions In: Proceedings of the 20th international conference on Computational Linguistics, page 1367.. Association for Computational Linguistics, Stroudsburg, PA, USA.

[11] Liu B (2010) Sentiment analysis and subjectivity In: Handbook of Natural Language Processing, Second Edition.. Taylor and Francis Group,

[12] Liu B, Hu M, Cheng J (2005) Opinion observer: Analyzing and comparing opinions on the web In: Proceedings of the 14th International Conference on World Wide Web, WWW ’05, 342–351.. ACM, New York, NY, USA.

[13] Pak A, Paroubek P (2010) Twitter as a corpus for sentiment analysis and opinion mining In: Proceedings of the Seventh conference on International Language Resources and Evaluation.. European Languages Resources Association, Valletta, Malta.

[14] Wilson T, Wiebe J, Hoffmann P (2005) Recognizing contextual polarity in phrase-level sentiment analysis In: Proceedings of the conference on human language technology and empirical methods in natural language processing, 347–354.. Association for Computational

Linguistics, Stroudsburg, PA, USA.

[15] Jindal N, Liu B (2008) Opinion spam and analysis In: Proceedings of the 2008 International Conference on, Web Search and Data Mining, WSDM ’08, 219–230.. ACM, New York, NY, USA.

[16] Mukherjee A, Liu B, Glance N (2012) Spotting fake reviewer groups in consumer reviews In: Proceedings of the 21st, International Conference on World Wide Web, WWW ’12, 191–200.. ACM, New York, NY, USA.

[17]https://www.google.com/imgres?imgurl=https%3A%2F%2Freview.chinabrands.com%2Fchinabrands%2Fseo%2Fimage%2F20181217%2F2018121705134014543361.jpg&imgrefurl=https%3A%2F%2Fwww.chinabrands.com%2Fdropshipping%2Farticle-e-commerce-websites-india-15499.html&tbnid=z0MLnBlT77IzPM&vet=12ahUKEwiu5PDJt__wAhUKTH0KHTAGDgwQMygBegUIARC3AQ

[18]https://www.google.com/imgres?imgurl=https%3A%2F%2Fwww.xoriant.com%2Fsites%2Fdefault%2Ffiles%2Fuploads%2F2020%2F01%2FBusiness_application.png&imgrefurl=https%3A%2F%2Fwww.xoriant.com%2Fblog%2Fmachine-learning%2Fnatural-language-processing-the-next-disruptive-technology-under-ai-part-i.html&tbnid=22lK9-Ksb30dTM&vet=12ahUKEwiY8emuP_wAhUVRysKHUWtCwkQMygaegUIARDsAQ

[19] https://www.techinasia.com/top-ecommerce-sites-india

[20] https://towardsdatascience.com/using-nlp-to-figure-out-what-people-really-think-e1d10d98e491

[21] https://searchenterpriseai.techtarget.com/definition/natural-language-processing-NLP

[22] https://www.semanticscholar.org/paper/A-Review-on-Natural-Language-Processing-in-Opinion-Bhattacharyya-Biswas/

[23] https://www.nltk.org/api/nltk.html#:~:text=data%20Module,grammars%2C%20and%20saved%20processing%20objects.&text=nltk%3Apath%20%3A%20Specifies%20the%20file,the%20directories%20specified%20by%20nltk.

[24] https://www.analyticsvidhya.com/blog/2021/06/part-9-step-by-step-guide-to-master-nlp-semantic-analysis/

[25]https://www.expert.ai/blog/natural-language-process-semantic-analysis-definition/