

1. Introduction

The popularity of mobile devices makes people's daily lives more dependent on mobile services. People get business information, product information, promotion information, and recommendation information from mobile devices. An important application of mobile services is movie recommendation. A movie recommender system has proven to be a powerful tool on providing useful movie suggestions for users. The suggestions are provided to support the users in their effort to cope with the information overload and help them find appropriate movies fast and conveniently. Different from the demand on the Personal Computers (PCs), mobile services place more emphasis on timeliness, which requires fast processing and calculation from service providers. Therefore, movie recommendation in mobile services needs to be promoted in both the recommendation accuracy and the timeliness.

Movie recommendation is a comprehensive and complicated task which involves various tastes of users, various genres of movies, and so forth. Therefore, various techniques for recommendation are planned to unravel the issues. For instance, content-based recommender system, collaborative filtering recommender system, and hybrid recommender system. Each technique has its own advantage in solving specific problems. Considering the usage of online information and user-generated content, collaborative filtering is supposed to be the most popular and widely deployed technique in recommender system. Collaborative filtering method recommends items by measuring the similarity between users. The similarity between users' preference can be measured by correlation calculation. In this way, users who have similar interest in movies are sorted in the same group, and then movies are recommended by their reviews and ratings of movies that they have seen. However, the correlation and similarity are difficult to calculate due to the sparsity of user's basic data, such as users' rating on movies that they have watched and their browsing history. Actually, the reviews of users on movies usually contain more information such as users' preference. Moreover, the content of sentiment that users have additionally a giant drawback in motion-picture show recommendation. At present, people are increasingly willing to post their own reviews online. In their reviews, users can express their preferences and feelings about movies. And the feelings contained in these reviews also affect the choice of other users. Users will see the reviews, analyze their personal experience, choose their useful reviews, remove some misleading or even harmful reviews, and ultimately make their own judgments and decisions. Therefore, the sentiment in reviews is a very important aspect in

evaluating a movie. Generally language, users are more inclined to choose the movies that the majority of people prefer and abandon the movies that the majority of people dislike. The decisions are made according to other people's experience to achieve users' own comfort experience.

Including the rise of the quantity of information, the way to give users with high quality recommendations quickly among the large data has become a significant drawback. The arrival of the mobile services makes the response speed an important indicator of the user experience. The text mining and sentiment analysis techniques used to deal with user reviews aggravate the difficulty of the recommender system in the traditional environment.

The proposed work, a sentiment-enhanced Natural Language Processing Framework method is proposed to recommend appropriate movies to users on Python platform. Sentiment analysis is more reliable than simple rating, due to the fact that it contains more emotional information, which proves to be powerful in arts items such as movies. Moreover, the high efficiency of Python (NLP) makes it possible to improve the timeliness of mobile services.

2. Literature Survey

2.1. Recommender Systems and Classifications

As the World Wide Web continues to grow at an exponential rate, the size and complexity of many web sites grow along with it. For the users of these web sites it becomes increasingly difficult and time consuming to find the information they are looking for. User interfaces could help users find the information that is in accordance with their interests by personalizing a web site. Some web sites present users with personalized information by letting them choose from a set of predefined topics of interest. Users however do not always know what they are interested in beforehand and their interests may change overtime which would require them to change their selection frequently. Recommender systems provide personalized information by learning the user's interests from traces of interaction with that user.

Recommender systems are classified into different categories based on the techniques. The following is a list of classifications:

2.1.1. Demographic recommender system

In these recommender systems the demographic aspects of users play a role in recommendation, like the age, language, country. The assumption is that recommendation should vary following the change in demographic. These solutions are common in the marketing domain. There is not a lot of work about them in recommender systems.

2.1.2. Knowledge-based recommenders

In these systems recommended items are based on the domain knowledge, answering questions, how some of the item features respond to user needs and preferences, as well as how useful the item is to the user. These knowledge recommenders are case based. A similarity function finds how much the user needs match the recommendations.

2.1.3. Constraint-based systems

This is another kind of knowledge-based recommendation system, the main difference between the two is how the solution is calculated. In the case based recommendation, the recommended items are based on the similarity metrics, while constraint-based recommendations predominantly exploit predefined knowledge bases that contain explicit rules about how to relate customer requirements with item features. Knowledge-based systems usually give better results in

the beginning of their work. In order to maintain good results they need to have learning algorithms, otherwise they can be surpassed by algorithms like CF ones.

2.1.4. Community-based recommenders

The main idea in the community-based recommendation is to recommend to the user based on the preference of his friends, following the epigram “Tell me who your friends are, and I will tell you who you are”. Works in this domain were made possible thanks to the widespread use of online social networks.

2.1.5. Contextual recommendation system

In this quite recommender system the results of advice vary in step with the context of the user. For example, in the temporal context, the clothes recommended in summer vary totally than those recommended in winter. For a social context, an example is that a film recommended to the user alone can vary from a film recommended to be seen with his girlfriend or with the family, recommending a restaurant for a Saturday night with friends, which would vary from a restaurant for a lunch during the week with co-workers.

2.2. Movie Recommender Systems

A recommender system is a program that predicts users’ preferences and recommends appropriate products or services to a specific user based on users’ information and products or services information. The research on recommender systems is started by Group Lens research team from the University of Minnesota. Their research object is a movie recommender system called Movielens. Early research is mainly focused on the content of the recommender system which analyzed the characteristics of the object itself to complete the recommendation task. However, this recommendation method can only be confined to content analysis, which makes researchers and practitioners invest great efforts in designing new recommender systems. Researchers have proposed recommender systems based on collaborative filtering, association rules, utility, knowledge, social network, multi objective programming, clustering, and other theories and techniques.

Researchers have also studied recommendations on mobile devices. Most of the research on mobile recommendation focuses on location-based services. For example, Zheng et al. utilized GPS trajectory data to solve mobile recommendation problems [1]. They proposed a user-centered collaborative location and activity filtering method based on user-location-activity

relations and collaborative filtering recommendation method. On the basis of this study, Zheng et al. came up with an algorithm using ranking-based collective tensor and matrix factorization (MF) to recommend activities to users [2].

2.3. Content-Based Recommendation

Content-based movie recommendation methods have been widely explored in the past few years. A content-based movie recommender system using ratings of the movies as the social information. The experiments proved that their methods were more flexible and accurate. What is more, Ono et al. employed Bayesian networks to construct users' movie preference models based on their context [3]. Obviously, a variety of methods were used to excavate features of users and movies to recommend appropriate movies. In addition to use new technologies to explore features, new perspectives are also explored to build accurate profiles of users and movies. For example, Szomszor et al. introduced semantic web to analyze folksonomy hidden in the movies to help users discover appropriate movies [4]. However, the design of effective profiles is always the bottleneck of content-based recommender systems. Both researchers and practitioners have made great efforts in designing a new recommendation method to avoid the shortcoming of content-based recommender systems. In the late 20th century, Armstrong et al. [5] proposed a personalized navigation system called "Web Watcher" at the AAAI (American Association for Artificial Intelligence). In August 1995, Henry Lieberman, from the Massachusetts Institute of Technology, presented a personalized navigation agent "Letizia" at the International Joint Conference on Artificial Intelligence (IJCAI). In 1997, AT&T Labs presented personalized recommendation systems based on collaborative filtering, called "PHOAKS" [6] and "Referral Web" [7]. In 1999, Tanja Joerding of Dresden University of Technology in Germany implemented a personalized e-commerce prototype system, "TELLIM" [8]. In 2001, IBM added personalized features to its e-commerce platform, "Websphere" [9], to enable businesses to develop personalized e-commerce sites. In 2003, Google made "AdWords" profitable which provided relevant ads through keywords that users searched for. In 2009, Overstock (US famous online retailers) began to use ChoiceStream Company's personalized banner advertising program, to run product ads in some high-traffic sites.

2.4. Collaborative Filtering-Based Recommendation

Collaborative filtering is used to make up for the shortcomings of content-based algorithm. The collaborative filtering algorithm was divided into parts for deep analysis in movie recommendation by Herlocker et al. [10]. In the process of recommendation, Koren found that users' preference changed over time, so he came up with a recommendation method using temporal dynamics to solve the problem [11]. What is more, Hofmann implemented Gaussian probabilistic latent semantic analysis in the collaborative filtering method on movie recommendation research [12]. Researchers invested great efforts by adding new technologies to improve the performance of collaborative filtering methods on movie recommendation and they achieved good results.

Collaborative filtering is a prevalent tool used in recommender systems [13]. Marlin came up with a collaborative filtering method based on ratings [14]. Salakhutdinov and Mnih proposed a collaborative filtering method, Probabilistic Matrix Factorization, which can handle large scale of dataset. At the same time, Salakhutdinov et al. employed Restricted Boltzmann Machines to improve the performance of collaborative filtering [15]. The experiment results showed that Restricted Boltzmann Machines outperformed singular value decomposition (SVD) on Netflix dataset. Moreover, Koren combined improved latent factor models and neighborhood models on Netflix dataset [16]. The latent factor model used is SVD while the neighborhood model is optimized on loss function. What is more, researchers also introduced other data mining methods to optimize the recommender systems. For example, Rendle proposed Factorization Machines (FM) which combine support vector machines (SVM) with factorization models [17]. Zhen et al., used the regularized MF used in Probabilistic Matrix Factorization (PMF) with tagging information of movies [18].

However, collaborative filtering method introduced new drawbacks in making up for some of the shortcomings of the content-based method. For example, the scalability of collaborative filtering is poor. When users produce new behavior, it is difficult for collaborative filtering to respond immediately. Therefore, both researchers and practitioners are inclined to hybridize collaborative filtering method and content-based method to solve the problem [35, 36]. For example, Debnath et al., presented a collaborative filtering and content-based movie recommender system [21]. In the content-based part of the hybrid system, the importance of the feature is expressed in a weighted manner. Nazim Uddin et al. proposed a diverse-item selection algorithm for optimizing the output of collaborative filtering method to improve the performance of hybrid

recommender system [22]. On the basis of integration of content based method and collaborative filtering method, Soni et al. joined the analysis of review based text mining algorithm, making the recommendation more accurate [23]. Moreover, Ling et al. employed a rating model with a topic model based on reviews to make accurate predictions [24]. As can be seen from the above studies, the hybrid recommender system can not only improve the efficiency, but also improve the scalability of movie recommendation. Therefore, a hybrid recommendation model is an appropriate method of movie recommendation.

2.5. Sentiment Analysis

Sentiment analysis is the process of analyzing, processing, summarizing, and reasoning the emotional text [25]. At present, the accuracy of emotional polarity analysis based on online commentary text is gradually increasing, but one of the problems existing in emotional analysis is the lack of indepth analysis and application of the influence of sentiment analysis.

Pang et al., used supervised learning method in machine learning to classify emotional polarity of the movie commentary text into positive one and negative one, by using the part of speech (POS) N-gram grammar (n-gram) and Maximum Entropy (ME) [26]. Turney implemented the unsupervised learning of machine learning to study the polarity of the text emotion [27]. He first used tags to extract the word pair from reviews and then used Pointwise Mutual Information and Information Retrieval (PMI-IR) method to calculate the similarity between the words in the text and words in the corpus to determine the emotional polarity of the text. The commentary data come from the online comment site <http://Epinions.com>. The method obtained an accuracy of 65.83% in the movie reviews dataset.

The polarity of reviews of the movies and other goods or services can be divided into positive, negative, and neutral. In general, the researchers believe that positive information has a positive effect while negative information has a negative effect [28]. Based on this conclusion, some studies introduced sentiment analysis into the user's reviews and obtained the polarity of the reviews. Ten movies with most positive information were recommended to users [29]. Sun et al. came up with a sentiment-aware social media recommender system [30].

2.6. Big Data Analytics for Recommendation

The scalability problem of recommender system also makes it harder for researchers and practitioners to provide users with convenient and efficient services. Many efforts have been taken to solve the problem. Parallel computing is one of the most prevalent solutions. Zhou et al. built a parallel Matlab platform to implement a movie recommender system with collaborative filtering method [31]. In parallel computing, the operation efficiency of recommendation algorithms is higher than that of single machine operation. The introduction of the distributed computing framework makes the efficiency of the recommender system improve qualitatively.

2.7. The Contribution of Our Work

As mentioned before, various recommendation models have been suggested as powerful tools for movie recommendation. Previous practitioners and academic researchers focus on the improvement of the recommendation performance by using the combination of recommendation models. However, they ignored that, with the increase of users and items to recommend, the computational overhead has heavily increased. Therefore, this work proposes a sentiment-enhanced movie recommendation framework based on Natural Language Processing in python platform to meet the requirement of mobile services in aspects of high timeliness. In our approach, the Natural Language Processing procedures are taken into consideration. Based on sentiment enhanced Natural Language Processing methods, the preliminary output is optimized by the analysis of the effect from positive, neutral and negative information. Finally, experiments are carried out to prove the performance of our proposed method.

2.8. Social Networks

Social networks are a fundamental part of the social media family sites. Online social networks like Facebook and Twitter continue to grow. The below figure shows the number of users using some social networks within a month in the United States only. Facebook has 141 million users and Twitter has 93 million users compared to a mere 8 million on Flickr and 1 million on opinions. Amazon is not considered as a social network but instead is a recommender system. However it has been added in order to compare data by using Fig 2.8.a and Fig 2.8.b.

People (United States) per Month

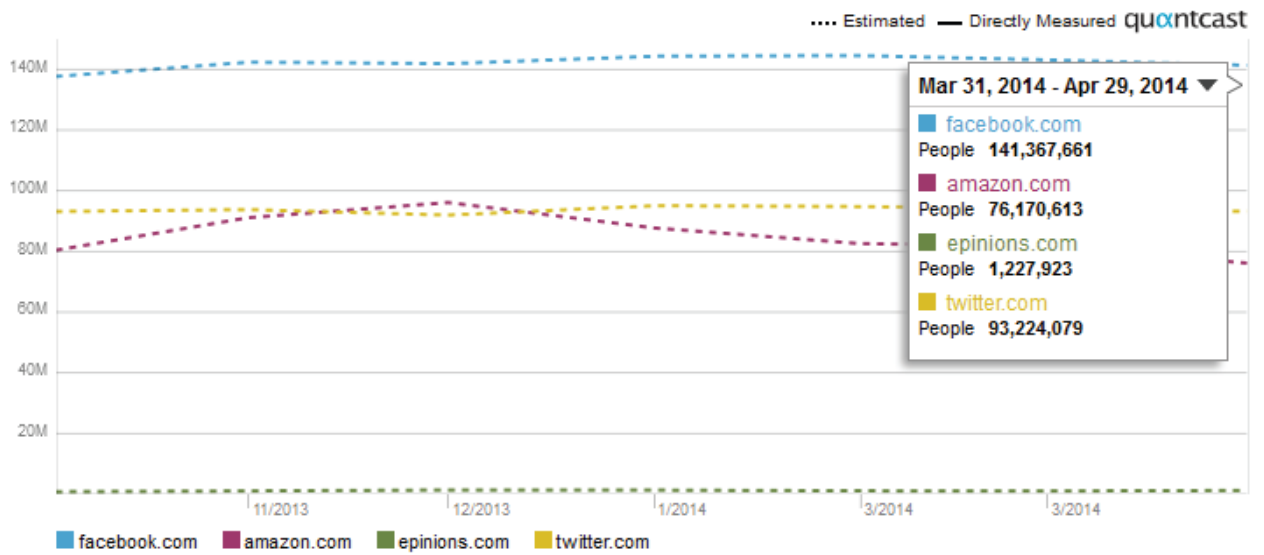


Fig 2.8.a: Comparing different sites over a month use in US (31-Mar-2014 – 29-Apr-2014) [14].

People (United States) per Month

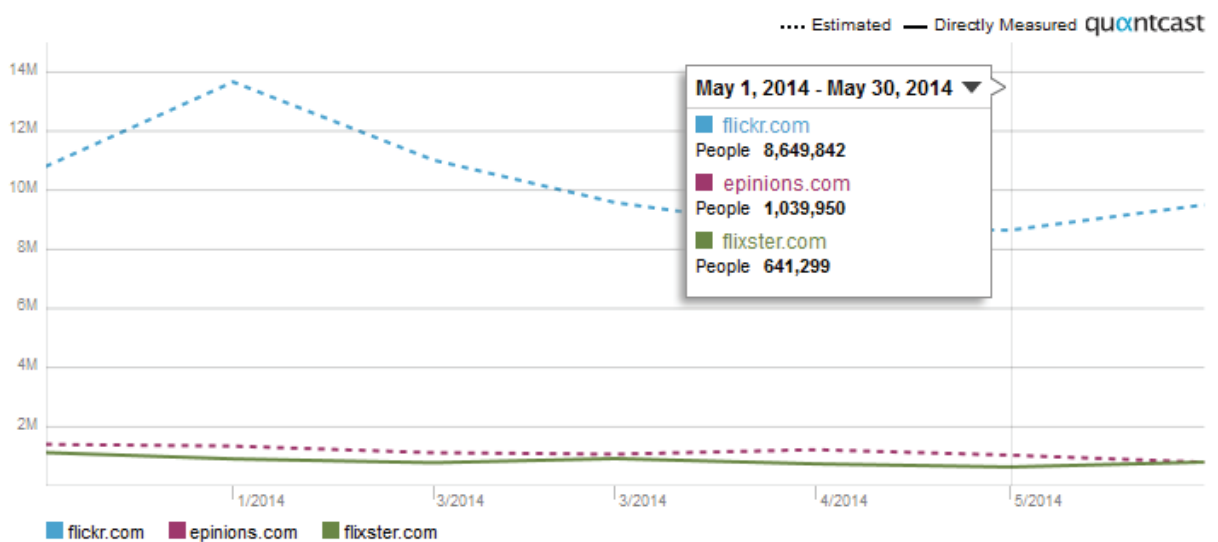


Fig 2.8.b: Comparing different sites over a month use in US (1-May-2014 – 30-May-2014) [14].

Online social networks are divided into General Purpose Social Networks GPSN like Twitter and Facebook, towards Domain Based Social Networks DSSN like opinions for product recommendation, Flickr for photo, LinkedIn for professional relations and so on.

2.8.1. General Purpose Social Networks GPSN

They are the social networks that do not have a domain associated with them (like films, products...). Users can do different actions in this kind of networks in different domains. Many GPSN exist like Classmates, MySpace, Face book and Twitter. The author claims that one main factor for which users use social networks is looking for the lost physical space around people, as people live in small homes and they are very busy, so they try to overcome this lack of physical space and time in social networks which are out of the space & time limits [32].

Facebook

Facebook was founded in 2004; one decade later, it has the global rank of 2nd and nearly 1.3 billion active users per month, 25% of which are from the United State. Since there are 7.2 billion humans on Earth, this would mean that for every five humans, one would have a Facebook account! In addition, 48% of these users log in every day and each user has an average of 130 friends. More than 50 percent of the users broadcast information and knowledge via Facebook. And Finally 70 percent log in every time they start their computer or web reader.

Facebook turned into a habit-forming activity and is a part of people's daily routine. Nearly half of the respondents announced that it is not easy to keep updated on top of things without Facebook [33]. Face booking (the use of Facebook) is sometimes considered as an unconscious habit. A majority of the respondents stated that they log in to their Facebook accounts every time they launch their web browser. Leif Denti, a doctoral student of Psychology at the University of Gothenburg stated that "this may even developed into an addiction."

The use of Facebook is somehow related to their commercial status, as people with low income and low-educated individuals spend more time on Facebook, and the worst fact is that women who use Facebook more are also report feeling less happy and less content with their lives. Facebook was not the first social network created but it is the most widely used to date. In Facebook, users can post phrases, comment on friends' post, and share videos and photos. They can also like items any kind, like books, films, restaurants. Additionally people can invite friends, and they can add all the places, countries and villages where the user has visited.

- Identity: In Facebook users can write posts, like items, do actions that can reflect their identity, and they can see directly the feedback of friends
- Relationship: Facebook permits to users to maintain their relation with their friends, while simultaneously seeking new relations.

- **Community:** Since the user has established his relations of old and new friends, he now has a community in which he can establish his social position. The basic desire is simple and age-old: to be recognized as a valued member of one's various communities.

Twitter

Twitter has nearly 645 million active users, 115 million are active every month, taking 5 days to produce 1 billion tweet! Twitter was founded in March 21, 2006 and is currently ranked #8 by Alexa.

Twitter is different from Facebook in the way in which it works; in Twitter a user can post short messages (limited to 140 characters), he can follow other users, as well as he can be followed by other users. This is how the community is created on Twitter. Furthermore the user can share photos and videos in addition to “re-tweeting”, replying, and “favoriting” tweets from other users.

Tweets can contain any information the user wishes to share. In general people use tweets to defuse information of what they are doing, where they are, how they are feeling, links to other sites, photos, or short videos, even for political and technical information.

Twitter reflects the trending topics instantly, not only by offering the trending topics but also by the participation of users who create tweets towards any news. Examples include the results of a sports match or an election and even news of Facebook being down for a few minutes.

The amount of information that exists on Twitter has caught the attention of researchers and they actually study these short messages.

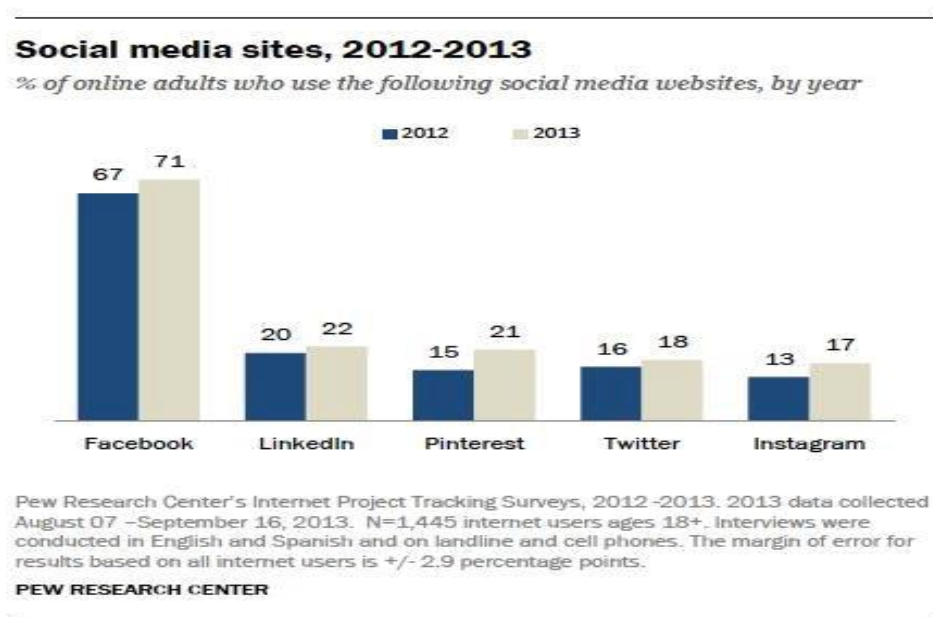
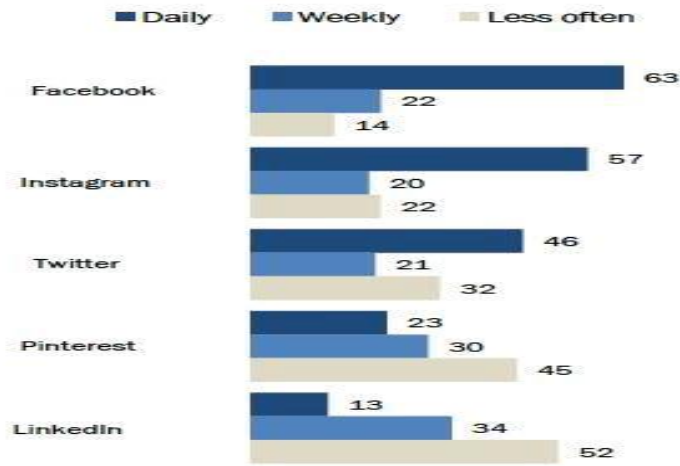


Fig 2.8.1.a: Social media sites usage adults evolution between 2012-2013, (63%) [33].

Frequency of social media site use

% of social media site users who use a particular site with the following frequencies (% is reported among each specific site's user groups, e.g., 63% of Facebook users use the site on a daily basis)



Pew Research Center's Internet Project August Tracking Survey, August 07 –September 16, 2013. Interviews were conducted in English and Spanish and on landline and cell phones.

PEW RESEARCH CENTER

Fig 2.8.1.b: Social media sites usage evolution between 2012-2013, (63%)[33].

3. System Analysis

3.1. Existing System

The existing system determines ratings based on reviews and the dataset is in Chinese language. In this system, both hybrid recommendation and sentimental analysis are used.

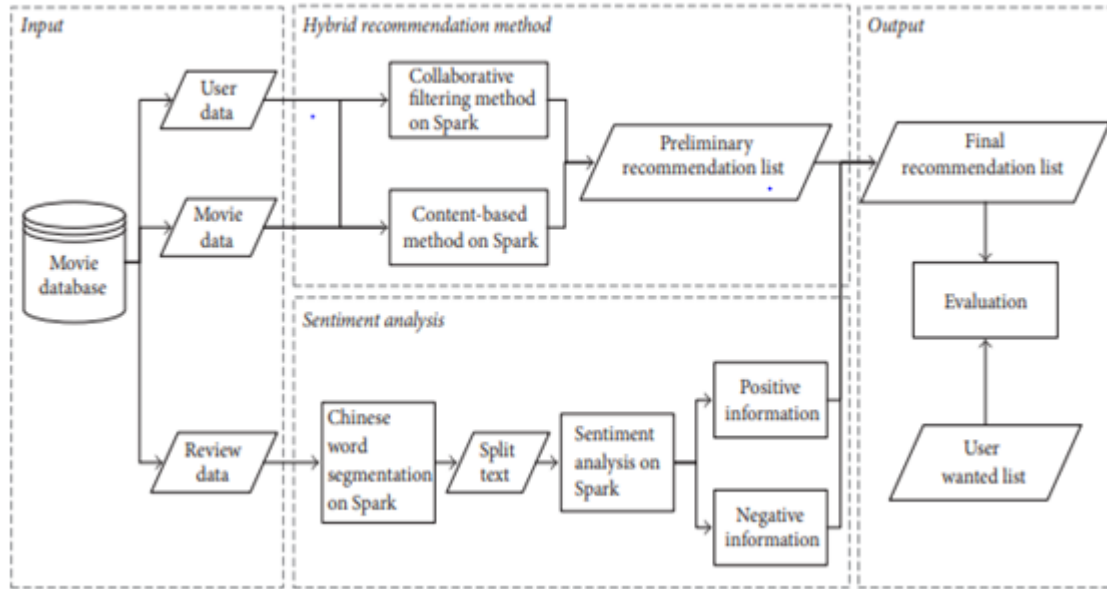


Fig 3.1: A sentiment-enhanced hybrid recommendation framework.

Problems with Existing System

This system uses both recommendation and sentiment analysis methods which takes more time to produce results.

The Existing System only determines ratings for the reviews which are in Chinese Language.

It takes more time to produce results and in Hybrid recommended systems there is a lack of knowledge problem.

3.2. Proposed System

The proposed system may contains only some Natural Language processing procedures which helps to classify(positive, neutral, negative) the words containing in the review. It more efficient when compared to existing system.

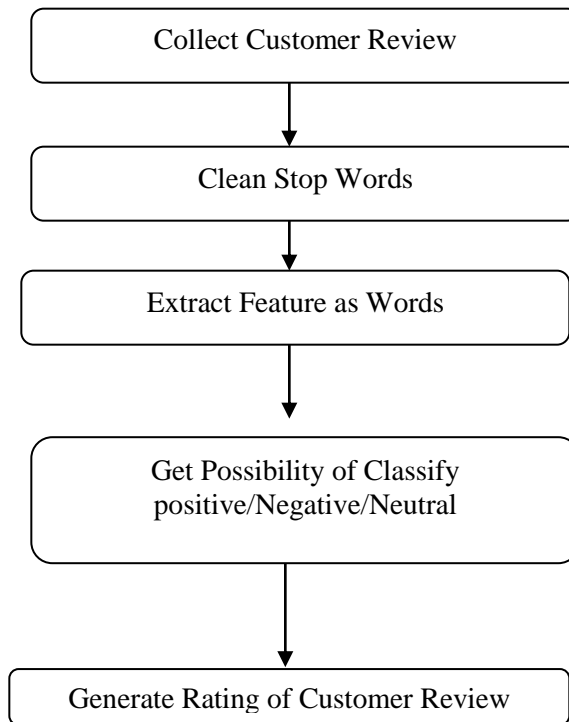


Fig 3.2: A sentiment enhanced NLP framework

Collect Reviews: Collect Reviews from Costumers which help us to predicate the nature of review i.e., positive , negative and neutral .

Clean/Remove Stop Words: Removing of meaning Less Word for example a, an, the, etc. By using Natural Language Processing Techniques called remove stop words.

Extract Feature as Words: Calculate the polarity for the words occurred in the filtered sentence which obtains after removing stop words. Polarity is used to classify the words as positive, negative and neutral.

If a word is said to be positive its polarity value must be greater than 0.5. If a word is said to be negative its polarity value must be less than -0.5. If a word is said to be neutral its polarity value may be in between -0.5 to 0.5.

Get Possibility of Classification: Classify the Word as Positive, Negative and Neutral. By using Natural Language Processing techniques like word tokenizes and Sentiment Intensity Analyzer.

Generate Rating: Generate rating based on words classified in the above step. Finally we draw a conclusion graphs by using Matplot library. Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc. It is used along with NumPy to provide an environment that is an effective open source alternative for MatLab. It can also be used with graphics toolkits like PyQt and wxPython.

Advantages of Proposed System

- This system uses only sentiment analysis methods which takes less time to produce results.
- The Proposed System determines ratings for the reviews which are in Universal Language.
- It takes less time to produce results by using NLP techniques where there is no use of Training datasets.

3.3. Feasibility Study

A Feasibility Study is a preliminary study undertaken before the real work of a project starts to ascertain the likely hood of the projects success. It is an analysis of possible alternative solutions to a problem and a recommendation on the best alternative.

- Economic Feasibility
- Technical Feasibility
- Operational Feasibility

Economic Feasibility

It is defined as the process of assessing the benefits and costs associated with the development of project. A proposed system, which is both operationally and technically feasible, must be a good investment for the organization.

With the proposed system the users are greatly benefited as the users can be able to detect the fake news from the real news and are aware of most real and most fake news published in the recent years. The proposed system does not need any additional software and high system configuration. Hence the proposed system is economically feasible.

Technical Feasibility

The technical feasibility infers whether the proposed system can be developed considering the technical issues like availability of the necessary technology, technical capacity, adequate response and extensibility.

The project is implemented using Python. Jupyter Notebook is designed for use in distributed environment of the internet and for the professional programmer it is easy to learn and use effectively. The developing organization has all the resources available to build the system therefore the proposed system is technically feasible.

Operational Feasibility

Operational feasibility is defined as the process of assessing the degree to which a proposed system solves business problems or takes advantage of business opportunities.

The system is self-explanatory and doesn't need any extra sophisticated training. The system has built in methods and classes which are required to produce the result. The application can be handled very easily with a novice user. The overall time that a user needs to get trained is less than one hour. As the software that is used for developing this application is very economical and is readily available in the market. Therefore the proposed system is operationally feasible.

3.4. Effort, Duration, and Cost Estimation using COCOMO Model

The COCOMO (Constructive Cost Model) model is the most complete and thoroughly documented model used in effort estimation. The model provides detailed formulas for determining the development time schedule, overall development effort, and effort breakdown by phase and activity as well as maintenance effort. COCOMO estimates the effort in person months of direct labor. The primary effort factor is the number of Source Lines Of Code (SLOC) expressed in thousands of delivered source instructions (KDSI).

There are two major types of SLOC measures: Physical SLOC (LOC) and Logical SLOC (LLOC). Specific definitions of these two measures vary, but the most common definition of physical SLOC is a count of lines in the text of the program's source code including comment lines. Blank lines are also included unless the lines of code in a section consists of more than 25% blank lines. In this case blank lines in excess of 25% are not counted toward lines of code.

Logical LOC attempts to measure the number of "statements", but their specific definitions are tied to specific computer languages (one simple Logical LOC measure for C-like programming languages is the number of statement-terminating semicolons). It is much easier to create tools that measure physical SLOC, and physical SLOC definitions are easier to explain. However, physical SLOC measures are sensitive to logically irrelevant formatting and style conventions, while logical LOC is less sensitive to formatting and style conventions. Unfortunately, SLOC measures are often stated without giving their definition, and logical LOC can often be significantly different from physical SLOC.

The model is developed in three versions of different level of detail basic, intermediate, and detailed. The overall modeling process takes into account three classes of systems.

Embedded: This class of system is characterized by tight constraints, changing environment, and unfamiliar surroundings. Projects of the embedded type are model to the company and usually exhibit temporal constraints.

Organic: This category encompasses all systems that are small relative to project size and team size, and have a stable environment, familiar surroundings and relaxed interfaces. These are simple business systems, data processing systems, and small software libraries.

Semidetached: The software systems falling under this category are a mix of those of organic and embedded in nature. Some examples of software of this class are operating systems, database management system, and inventory management systems.

For Basic COCOMO

$$\text{Effort} = a * (\text{KLOC})^b$$

$$\text{Type} = c * (\text{effort})^d$$

For Intermediate and Detailed COCOMO

$$\text{Effort} = a * (\text{KLOC})^b * \text{EAF} \quad (\text{EAF} = \text{product of cost drivers})$$

Type of Product	A	B	C	D
Organic	2.4	1.02	2.5	0.38
Semi Detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Table 3.4: The table gives four values for Organic, Semidetached and Embedded systems.

Intermediate COCOMO model is a refinement of the basic model, which comes in the function of 15 attributes of the product. For each of the attributes the user of the model has to provide a rating

using the following six point scale.

VL (Very Low)

HI (High)

LO (Low)

VH (VeryHigh)

NM (Nominal)

XH (ExtraHigh)

The list of attributes is composed of several features of the software and includes product, computer, personal and project attributes as follows.

Product Attributes

Required reliability (RELY): It is used to express an effect of software faults ranging from slight inconvenience (VL) to loss of life (VH). The nominal value (NM) denotes moderate recoverable losses.

Data bytes per DSI (DATA): The lower rating comes with lower size of a database.

Complexity (CPLX): The attribute expresses code complexity again ranging from straight batch code (VL) to real time code with multiple resources scheduling (XH).

Computer Attributes:

Execution Time (TIME) and memory (STOR) constraints: This attribute identifies the percentage of computer resources used by the system. NM states that less than 50% is used; 95% is indicated by XH.

Virtual machine volatility (VIRT): It is used to indicate the frequency of changes made to the hardware, operating system, and overall software environment. More frequent and significant changes are indicated by higher ratings.

Development turnaround time (TURN): This is a time from when a job is submitted until output becomes received. LO indicated a highly interactive environment, VH quantifies a situation when this time is longer than 12 hours.

Personal attributes

Analyst capability (ACAP) and programmer capability (PCAP): Describe skills of the developing team. The higher the skills the higher the rating.

Application experience (AEXP), language experience (LEXP), and virtual machine experience (VEXP): These are used to quantify the number of experience in each area by the development team, more experience, higher rating.

Project attributes

Modern Development Practices (MODP): Deals with the amount of use of modern software practices such as structural programming and object oriented approach.

Use of software tools (TOOL): This is used to measure a level of sophistication of automated tools used in software development and a degree of integration among the tools being used. Higher rating describes levels in both aspects.

Schedule effects (SCED): concerns the amount of schedule compression (HI or VH), or schedule expansion (LO or VL) of the development schedule in comparison to a nominal (NM) schedule.

	VL	LO	NM	HI	VH	XH
RELY	0.75	0.88	1.00	1.15	1.40	
DATA		0.94	1.00	1.08	1.16	
CPLX	0.70	0.85	1.00	1.15	1.30	1.65
TIME			1.00	1.11	1.30	1.66
STOR			1.00	1.06	1.21	1.56
VIRT		0.87	1.00	1.15	1.30	
TURN		0.87	1.00	1.15	1.30	
ACAP	1.46	1.19	1.00	0.86	0.71	
AEXP	1.29	1.13	1.00	0.91	0.82	
PCAP	1.42	1.17	1.00	0.86	0.70	
LEXP	1.14	1.07	1.00	0.95		
VEXP	1.21	1.10	1.00	0.90		
MODP	1.24	1.10	1.00	0.91	0.82	
TOOL	1.24	1.10	1.00	0.91	0.83	
SCED	1.23	1.08	1.00	1.04	1.10	

Table 3.4.b: Project rating using the six point scale.

Our project is an organic system and for intermediate COCOMO

$$\text{Effort} = a * (\text{KLOC})^b * \text{EAF}$$

$$\text{KLOC} = 115$$

$$\text{For organic system } a = 2.4$$

$$b = 1.02$$

EAF = product of cost drivers

$$\text{Effort} = 2.4 * (0.115)^{1.02} * 1.30$$

$$= 1.034 \text{ programmer months}$$

$$\text{Time for development} = C * (\text{Effort})^d$$

$$= 2.5 * (1.034)^{0.38}$$

$$= 2.71 \text{ months}$$

Cost of programmer = Effort * cost of programmer per month

$$= 1.034 * 20000$$

$$= 20680$$

Project cost = 20000 + 20680

$$= 40680$$

4. Software Requirement Specification

A Software Requirements Specification (SRS) is a description of a particular software product, program or set of programs that performs a set of functions in a target environment (IEEE Std. 830-1993).

4.1.1. Purpose

The purpose of software requirements specification specifies the intentions and intended audience of the SRS.

4.1.2. Scope

The scope of the SRS identifies the software product to be produced, the capabilities, application, relevant objects etc.

4.1.3. Definitions, Acronyms and Abbreviations

Software Requirements Specification: It's a description of a particular software product, program or set of programs that performs a set of function in target environment.

4.1.4. References

IEEE Std. 830-1993, IEEE Recommended Practice for Software Requirements Specifications. Software Engineering by James F Peters & Witold Pedrycz. Head First Java by Kathy Sierra and Bert Bates.

4.1.5. Overview

The SRS contains the details of process, DFD's, functions of the product, user characteristics. The non-functional requirements if any are also specified.

4.2. Overall Description

The main functions associated with the product are described in this section of SRS. The characteristics of a user of this product are indicated. The assumptions in this section result from interaction with the project stakeholders.

Requirement Analysis

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase). Under requirement specification, the focus is on specifying what has been found giving analysis such as

representation, specification languages and tools, and checking the specifications are addressed during this activity.

The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase. The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium through which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

4.2.1. Product Perspective

The application is developed in such a way that any future enhancement can be easily implementable. The project is developed in such a way that it requires minimal maintenance. The software used are open source and easy to install. The application developed should be easy to install and use.

This is an independent application which can be easily run on to any system which has Python installed and Jupyter Notebook.

4.2.1.1. Product features

The application is developed in such a way that Analysis the sentiment using Natural Language Processing Techniques. The data is taken from the users or customers. We can predict the ratings using NLP techniques like stop words and sentiment analyzers.

4.2.1.2. User characteristics

Application is developed in such a way that its users are

- Easy to use
- Error free
- Minimal training or no training
- Patient regular monitor

4.2.1.3. Assumption & Dependencies

It is considered that the dataset taken fulfils all the requirements.

4.2.1.4. Domain Requirements

This document is the only one that describes the requirements of the system. It is meant for the use by the developers, and will also be the basis for validating the final delivered system. Any changes made to the requirements in the future will have to go through a formal change approval process.

4.2.1.5. User Requirements

User can decide on the prediction accuracy to decide on which algorithm can be used in real-time predictions.

4.2.2. Non Functional Requirements

- Dataset collected should be in the JSON format
- The attribute values should be any values
- Error rates can be calculated for prediction algorithm.

4.2.2.1. Implementation Requirements

The dataset collection, internet connection to install related libraries.

4.2.2.2. Engineering Standard Requirements

User Interfaces

User interface is developed in python, which gets input such stock symbol.

Hardware Interfaces

Ethernet

Ethernet on the AS/400 supports TCP/IP, Advanced Peer-to-Peer Networking (APPN) and advanced program-to-program communications (APPC).

ISDN

To connect AS/400 to an Integrated Services Digital Network (ISDN) for faster, more accurate data transmission. An ISDN is a public or private digital communications network that can support data, fax, image, and other services over the same physical interface. can use other protocols on ISDN, such as IDLC and X.25.

Software Interfaces

Jupyter Notebook is used to develop this project. Jupyter Notebook is used to implement the project in python in an effective manner.

4.2.2.3. Operational Requirements

Economic

The developed product is economic as it is not required any hardware interface etc.

Environmental

Statements of fact and assumptions that define the expectations of the system in terms of mission objectives, environment, constraints, and measures of effectiveness and suitability (MOE/MOS). The customers are those that perform the eight primary functions of systems

engineering, with special emphasis on the operator as the key customer.

Health and Safety

The software may be safety-critical. If so, there are issues associated with its integrity level. The software may not be safety-critical although it forms part of a safety-critical system. For example, software may simply log transactions. If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level. There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable. If a computer system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level. Systems with different requirements for safety levels must be separated. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

4.2.3. System Requirements

4.2.3.1. Hardware Requirements

Processor	: Any Processor above 500 MHz.
Ram	: 4 GB
Hard Disk	: 4 GB
Input device	: Standard Keyboard and Mouse.
Output device	: VGA and High Resolution Monitor.

4.2.3.2. Software Requirements

Operating System	: Windows 7 or higher
Programming	: Python 3.6 and related libraries
Software	: Jupyter NoteBoook

5. System Design

5.1. System Architecture

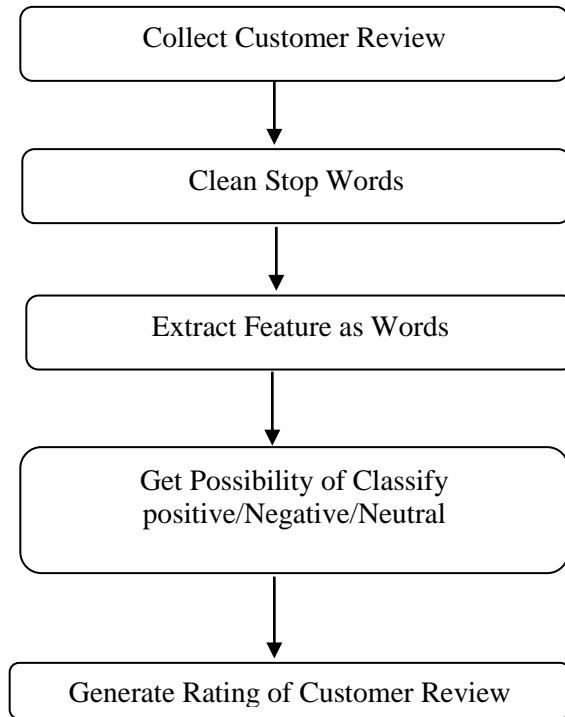


Fig 5.1: System Architecture for a sentiment enhanced NLP framework

Collect Reviews: Collect Reviews from Costumers which help us to predicate the nature of review i.e., positive, negative and neutral .

Clean/Remove Stop Words: Removing of meaning Less word for example a, an, the, etc. By using Natural Language Processing Techniques called remove stop words.

Extract Feature as Words: Calculate the polarity for the words occurred in the filtered sentence which obtains after removing stop words. Polarity is used to classify the words as positive, negative and neutral.

If a word is said to be positive its polarity value must be greater than 0.5. If a word is said to be negative its polarity value must be less than -0.5. If a word is said to be neutral its polarity value may be in between -0.5 to 0.5.

Get Possibility of Classification: Classify the word as Positive, Negative and Neutral. By using Natural Language Processing techniques like word tokenizes and Sentiment Intensity Analyzer.

Generate Rating: Generate rating based on words classified in the above step. Finally we draw a conclusion graphs by using Matplot library. Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc. It is used along with NumPy to provide an environment that is an effective open source alternative for MatLab.

5.2. Data Flow Diagram

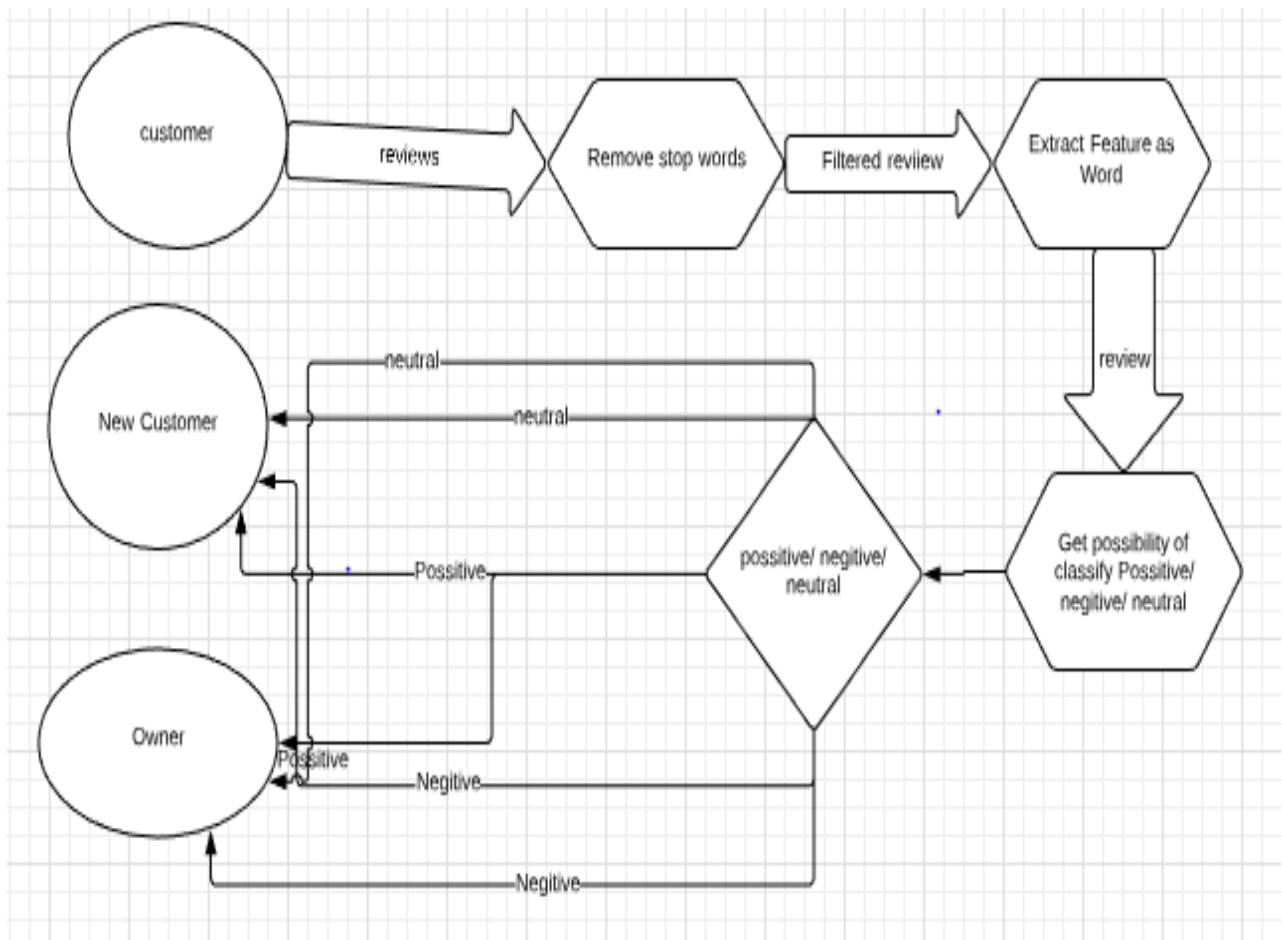


Fig 5.2: Data-flow diagram for a sentiment enhanced NLP framework.

Customer: Customer is a person who buys the item, watches the movie, listen the music, etc. Customer gives the review for a particular item.

New Customer: New Customer is a person who wants to buy the items, watches the movie, listen the music in future based on the previous reviews or ratings given by the customer.

Owners: Owner is an actor who knows the ratings and improves the quality of an item based on ratings, by taking certain measures and precautions.

Clean/Remove Stop Words: Removing of meaning Less word for example a, an, the, etc. By using Natural Language Processing Techniques called remove stop words.

Extract Feature as Words: Calculate the polarity for the words occurred in the filtered sentence which obtains after removing stop words. Polarity is used to classify the words as positive,

negative and neutral.

If a word is said to be positive its polarity value must be greater than 0.5. If a word is said to be negative its polarity value must be less than -0.5. If a word is said to be neutral its polarity value may be in between -0.5 to 0.5.

Get Possibility of Classification: Classify the word as Positive, Negative and Neutral. By using Natural Language Processing techniques like word tokenizes and Sentiment Intensity Analyzer.

5.3. Flow chart

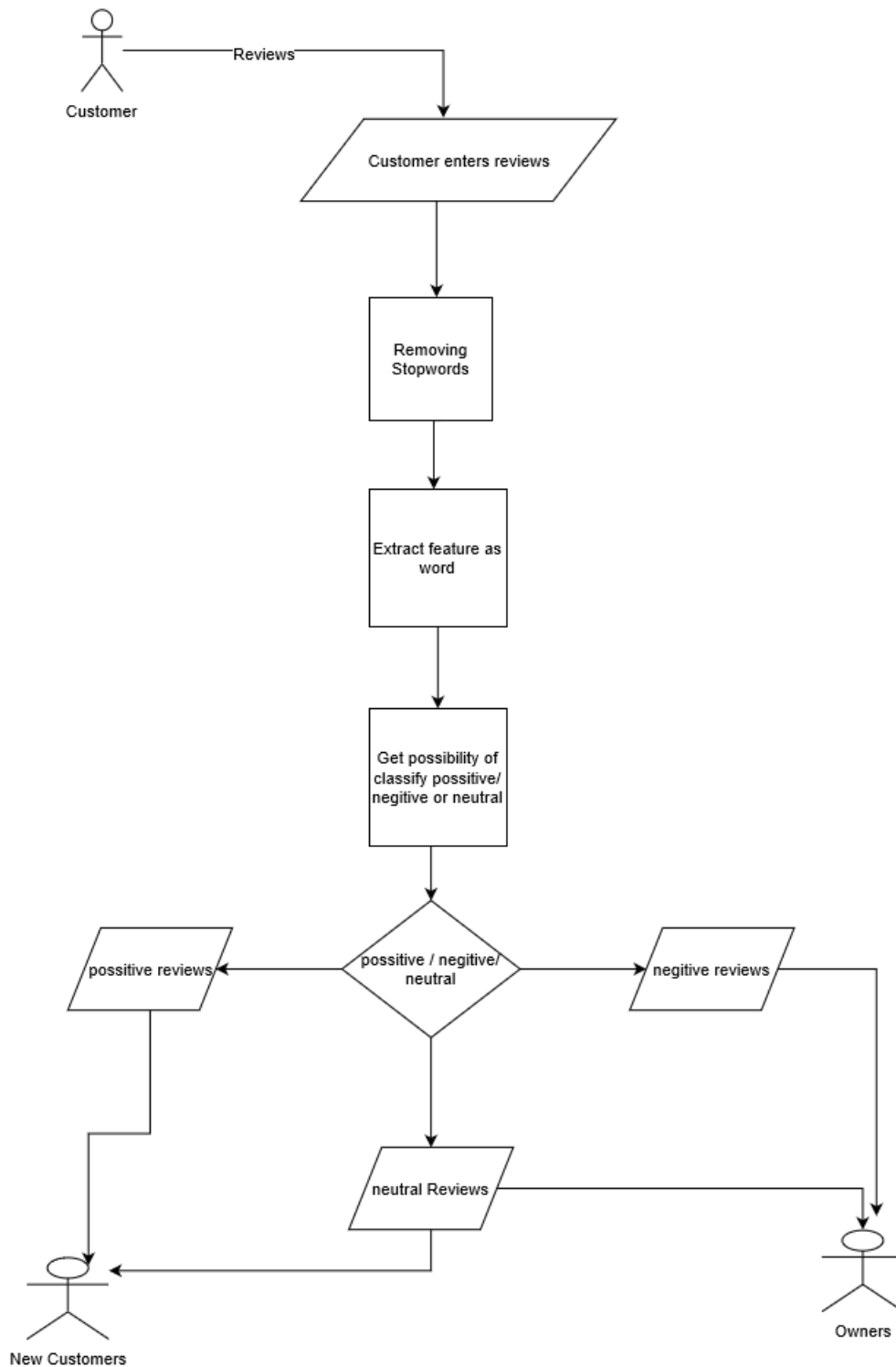


Fig 5.3: Flow chart for a sentiment enhanced NLP framework

Customer: Customer is a person who buys the item, watches the movie, listen the music, etc. Customer gives the review for a particular item.

New Customer: New Customer is a person who wants to buy the items, watches the movie, listen the music in future based on the previous reviews or ratings given by the customer.

Owners: Owner is an actor who knows the ratings and improves the quality of an item based on ratings, by taking certain measures and precautions.

Clean/Remove Stop Words: Removing of meaning Less word for example a, an, the, etc. By using Natural Language Processing techniques called remove stop words.

Extract Feature as Words: Calculate the polarity for the words occurred in the filtered sentence which obtains after removing stop words. Polarity is used to classify the words as positive, negative and neutral.

If a word is said to be positive its polarity value must be greater than 0.5. If a word is said to be negative its polarity value must be less than -0.5. If a word is said to be neutral its polarity value may be in between -0.5 to 0.5.

Get Possibility of Classification: Classify the word as Positive, Negative and Neutral. By using Natural Language Processing techniques like tokenizes and Sentiment Intensity Analyzer. And these ratings are viewed by other customers (willing to buy or willing to watch a movie) and owners (for providing more efficiency and performance).

5.4. UML Diagrams

5.4.1. Use case Diagram

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The relationships between and among the actors and the use cases. The fig 5.4.1. represents the Use case diagram for a sentiment enhanced NLP framework.

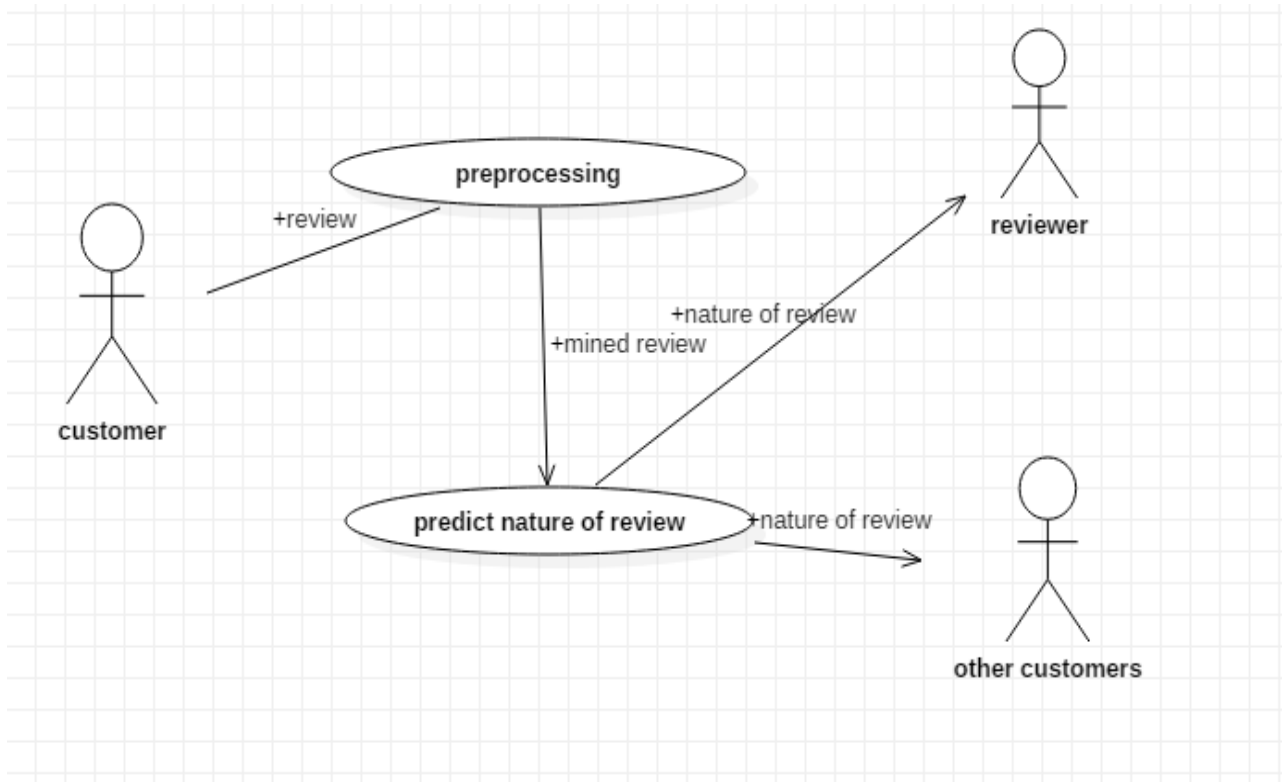


Fig 5.4.1: Use case diagram for a sentiment enhanced NLP framework

Customer: Customer is a person who buys the item, watches the movie, listen the music, etc. Customer gives the review for a particular item.

New Customer: New Customer is a person who wants to buy the items, watches the movie, listen the music in future based on the previous reviews or ratings given by the customer.

Owners: Owner is an actor who knows the ratings and improves the quality of an item based on ratings, by taking certain measures and precautions.

Preprocessing: Removing of meaning Less word for example a, an, the, etc. By using Natural Language Processing Techniques called remove stop words.

Predict Nature of Review: Calculate the polarity for the words occurred in the filtered sentence which obtains after removing stop words. Classify the word as Positive, Negative and Neutral. By using Natural Language Processing techniques like word tokenizes and Sentiment Intensity Analyzer based on polarity. It is used to classify the words as positive, negative and neutral. If a word is said to be positive its polarity value must be greater than 0.5. If a word is said to be negative its polarity value must be less than -0.5 If a word is said to be neutral its polarity value may be in between -0.5 to 0.5. Generate rating based on words classified in the above step. Finally we draw a conclusion graphs by using Matplot library.

5.4.2. Class Diagram

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modeling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity. Class diagrams are useful in all forms of object-oriented programming (OOP). The concept is several years old but has been refined as OOP modeling paradigms have evolved. The fig 5.4.2 represents the class diagram for a sentiment enhanced NLP framework.

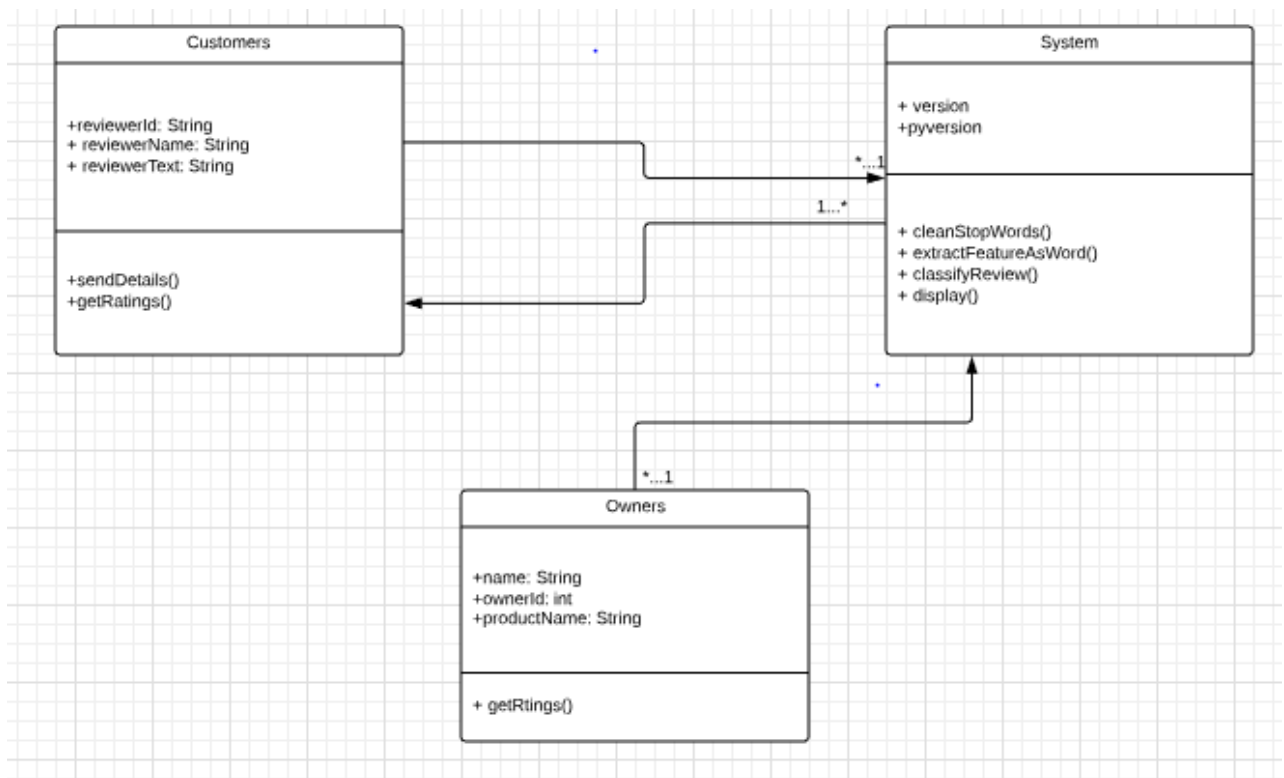


Fig 5.4.2: Class diagram for a sentiment enhanced NLP framework

Customer: Customer is a person who buys the item, watches the movie, listen the music, etc. Customer gives the review for a particular item.

New Customer: New Customer is a person who wants to buy the items, watches the movie, listen the music in future based on the previous reviews or ratings given by the customer.

System

Clean/Remove Stop Words: Removing of meaning Less word for example a, an, the, etc. By using Natural Language Processing techniques called remove stop words.

Extract Feature as Words: Calculate the polarity for the words occurred in the filtered sentence which obtains after removing stop words. Polarity is used to classify the words as positive, negative and neutral.

If a word is said to be positive its polarity value must be greater than 0.5. If a word is said to be negative its polarity value must be less than -0.5. If a word is said to be neutral its polarity value may be in between -0.5 to 0.5.

Get Possibility of Classification: Classify the word as Positive, Negative and Neutral. By using Natural Language Processing techniques like word tokenizes and Sentiment Intensity Analyzer.

Generate Rating : Generate rating based on words classified in the above step. Finally we draw a conclusion graphs by using Matplot library. Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc. It is used along with NumPy to provide an environment that is an effective open source alternative for MatLab.

Owners: Owner is an actor who knows the ratings and improves the quality of an item based on ratings, by taking certain measures and precautions.

5.4.3. Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. The fig [5.4.3] represents the sequence diagram for a sentiment enhanced NLP framework.

Customer: Customer is a person who buys the item, watches the movie, listen the music, etc. Customer gives the review for a particular item.

New Customer: New Customer is a person who wants to buy the items, watches the movie, listen the music in future based on the previous reviews or ratings given by the customer.

System

Clean/Remove Stop Words: Removing of meaning Less word for example a, an, the, etc. By using Natural Language Processing Techniques called remove stop words.

Extract Feature as Words: Calculate the polarity for the words occurred in the filtered sentence which obtains after removing stop words. Polarity is used to classify the words as positive, negative and neutral.

If a word is said to be positive its polarity value must be greater than 0.5. If a word is said to be negative its polarity value must be less than -0.5. If a word is said to be neutral its polarity value may be in between -0.5 to 0.5.

Get Possibility of Classification: Classify the word as Positive, Negative and Neutral. By using Natural Language Processing techniques like word tokenizes and Sentiment Intensity Analyzer.

Generate Rating : Generate rating based on words classified in the above step. Finally we draw a conclusion graphs by using Matplot library. Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts .

Owners: Owner is an actor who knows the ratings and improves the quality of an item based on ratings, by taking certain measures and precautions.

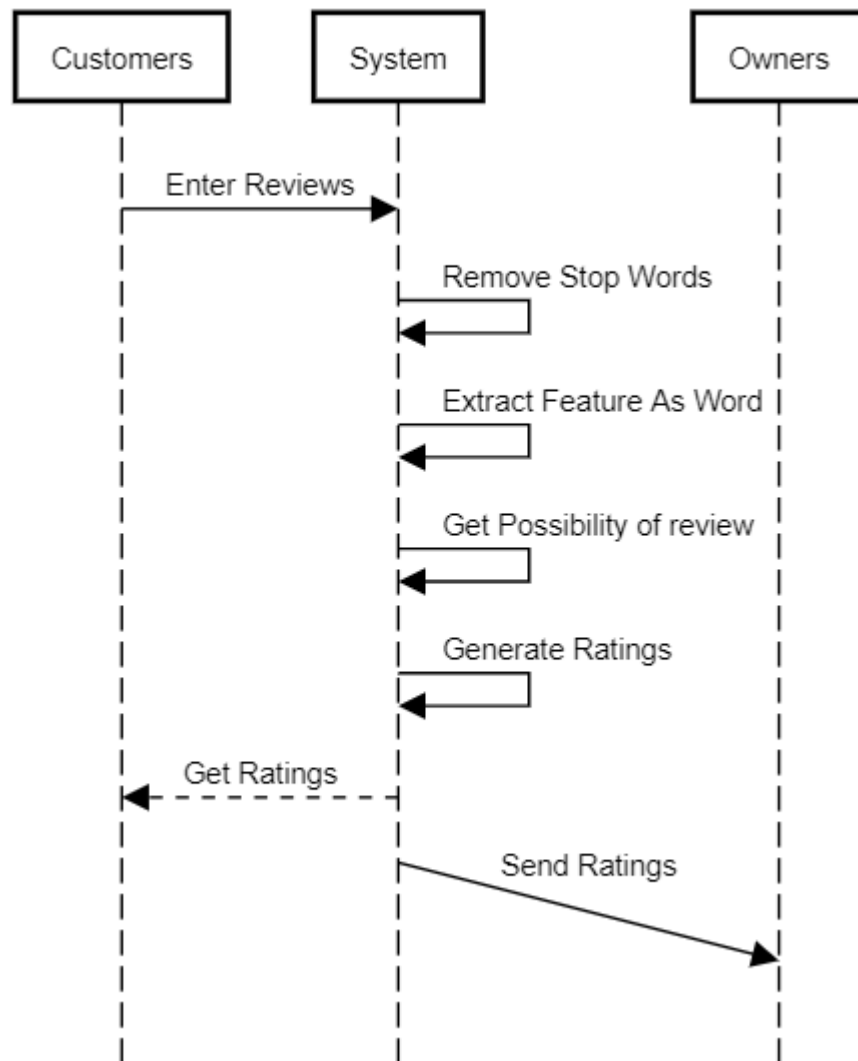


Fig 5.4.3: Sequence diagram for a sentiment enhanced NLP framework

6. Implementation

6.1. Algorithm

```
algorithm GenerateRatings()
begin
    read the input from the customer;
    gather review text from the given input;
    filtered_sentence := remove all the stop words;
    extract feature as word (calculating polarity for each word obtained in above step);
    classify the nature of the review (positive/ negative/ neutral);
    display the nature of review obtained in above step;
end
```

6.2. Analysis of an algorithm

In theoretical analysis of algorithms, it is common to estimate their complexity in the asymptotic sense, i.e., to estimate the complexity function for arbitrarily large input. The term "analysis of algorithms" was coined by Donald Knuth.

Algorithm analysis is an important part of computational complexity theory, which provides theoretical estimation for the required resources of an algorithm to solve a specific computational problem. Most algorithms are designed to work with inputs of arbitrary length. Analysis of algorithms is the determination of the amount of time and space resources required to execute it.

We have two types of methods for finding analysis of an algorithm they are

1. Time complexity
2. Space complexity

Execution time of an algorithm depends on the instruction set, processor speed, disk I/O speed, etc. Hence, we estimate the efficiency of an algorithm asymptotically. Here is the clear explanation regarding Time complexity and Space complexity of an algorithm.

1. Time complexity: The efficiency or running time of an algorithm is stated as a function relating the input length to the number of steps.

- **Best case:** The minimum number of steps taken to solve an algorithm.

- **Average case:** An average number of steps taken to solve an algorithm.
- **Worst case:** maximum number of steps taken to solve an algorithm
- **Amortized** – A sequence of operations applied to the input averaged over time.

Algorithm contains more lines of code, it doesn't mean that it takes lot of time to execute an algorithm. But it depends on number of iterations does an algorithm executes.

The time complexity of an algorithm is same in all the three cases i.e., best, Average, Worst cases is $O(n)$.

Where 'n' depends on number of words does an input contain.

2. Space complexity: volume of memory occupied by an algorithm. It's a function describing the amount of memory an algorithm takes in terms of the size of input to the algorithm. We often speak of "extra" memory needed, not counting the memory needed to store the input itself. Again, we use natural (but fixed-length) units to measure. space complexity is sometimes ignored because the space used is minimal and/or obvious, however sometimes it becomes as important an issue as time.

Space complexity is ignored in this algorithm because the space used is minimal, obvious, and negligible.

6.3. Sample Code

```
import json

config = json.loads(open('C:/Users/Administrator/Desktop/jsonex.json').read())

l=config[0]

l['reviewText']

import nltk

from nltk.corpus import stopwords

from nltk.tokenize import word_tokenize

from nltk.tokenize import sent_tokenize, word_tokenize

example_sent =l['reviewText']

from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))

word_tokens = word_tokenize(example_sent)

filtered_sentence = [w for w in word_tokens if not w in stop_words]

filtered_sentence = []

for w in word_tokens:

    if w not in stop_words:

        filtered_sentence.append(w)

from nltk.sentiment.vader import SentimentIntensityAnalyzer

test_subset=filtered_sentence

sid = SentimentIntensityAnalyzer()

pos_word_list=[]

neu_word_list=[]

neg_word_list=[]
```

```

for word in test_subset:

    if (sid.polarity_scores(word)['compound']) >= 0.5:

        pos_word_list.append(word)

    elif (sid.polarity_scores(word)['compound']) <= -0.5:

        neg_word_list.append(word)

    else:

        neu_word_list.append(word)

from collections import Counter

Counter = Counter(neu_word_list)

most_occur = Counter.most_common()

lis = []

lis2 = []

length=len(most_occur)

length

for i in range(0,length):

    p=most_occur[i]

    lis.append(p[1])

for i in range(0,length):

    p=most_occur[i]

    lis2.append(p[0])

from collections import Counter

count = Counter(pos_word_list)

most_occur1 = count.most_common()

from collections import Counter

```



```

coun = Counter(neg_word_list)

most_occur2 = coun.most_common()

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

import numpy as np

import matplotlib.pyplot as plt

objects = tuple(lis2)#lis2 words

y_pos = np.arange(len(objects))

performance = lis#lis values

plt.bar(y_pos, performance, align='center', alpha=0.5)

plt.xticks(y_pos, objects)

plt.xlabel('Usage')

plt.title('words count')

plt.show()

lis3 = []

lis4 = []

length=len(most_occur1)

length

for i in range(0,length):

    q=most_occur1[i]

    lis3.append(q[1])

for i in range(0,length):

    q=most_occur1[i]

    lis4.append(q[0])

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

```

```

import numpy as np

import matplotlib.pyplot as plt

objects = tuple(lis4)#lis4 words

y_pos = np.arange(len(objects))

performance = lis3#lis3 values

plt.bar(y_pos, performance, align='center', alpha=0.5)

plt.xticks(y_pos, objects)

plt.xlabel('Usage')

plt.title('words count')

plt.show()

lis5 = []

lis6 = []

length=len(most_occur2)

length

for i in range(0,length):

    q=most_occur2[i]

    lis5.append(q[1])

for i in range(0,length):

    q=most_occur2[i]

    lis6.append(q[0])

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

import numpy as np

import matplotlib.pyplot as plt

objects = tuple(lis6)#lis4 words

```

```
y_pos = np.arange(len(objects))  
performance = lis5#lis3 values  
plt.bar(y_pos, performance, align='center', alpha=0.5)  
plt.xticks(y_pos, objects)  
plt.xlabel('Usage')  
plt.title('words count')  
plt.show()
```

7. Testing

Testing is a process of executing a program with the aim of finding error. To make our software perform well it should be error free. If testing is done successfully it will remove all the errors from the software.

Principles of Testing:

The entire test should meet the customer requirements. To make our software testing should be performed by third party. Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk assessment of the application. All the test to be conducted should be planned before implementing it. Start testing with small parts and extend it to large parts.

7.1. Black Box Testing

Black Box Testing also known as Behavioral Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

Input1 : Json file(.json ext)

```
{  
  
"reviewerID":    "A14R9XMZVJ6INB",    "asin":    "616719923X",    "reviewerName":  
"amf0001","helpful": [0, 1], "reviewText": "better to have option negotiation", "overall": 3.0,  
"summary": "3.5 stars, sadly not as wonderful as I had hoped", "unixReviewTime": 1400457600,  
"reviewTime": "05 19, 2014"  
}
```

Output :

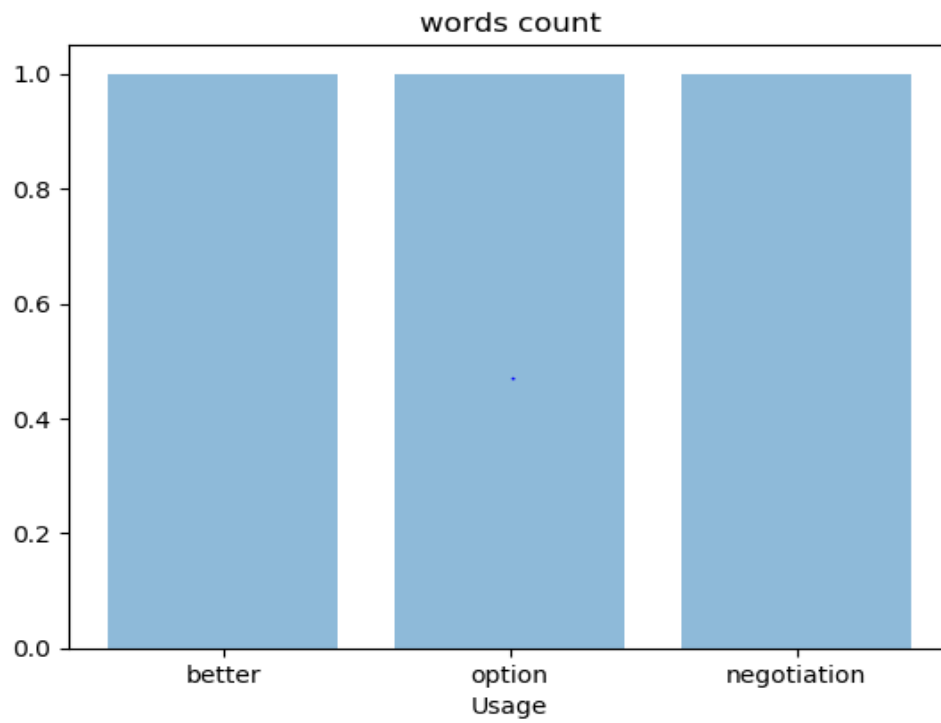


Fig 7.1.a: Neutral words in a review

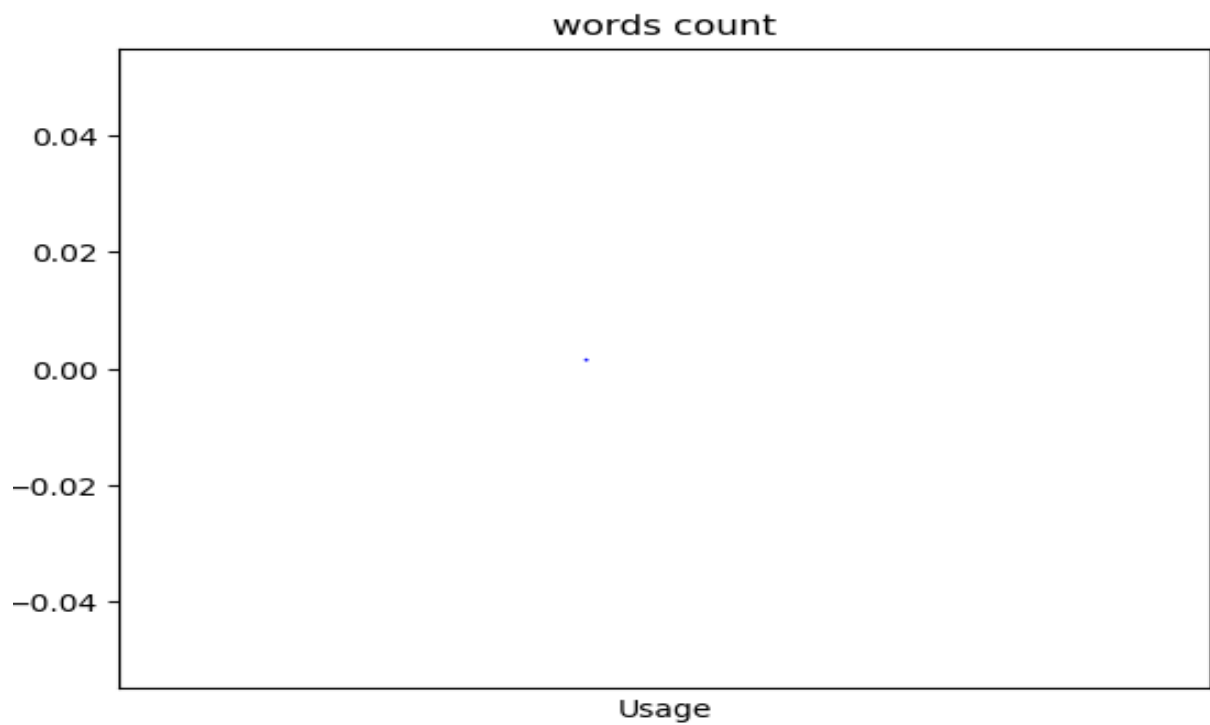


Fig 7.1.b: Positive words in a review

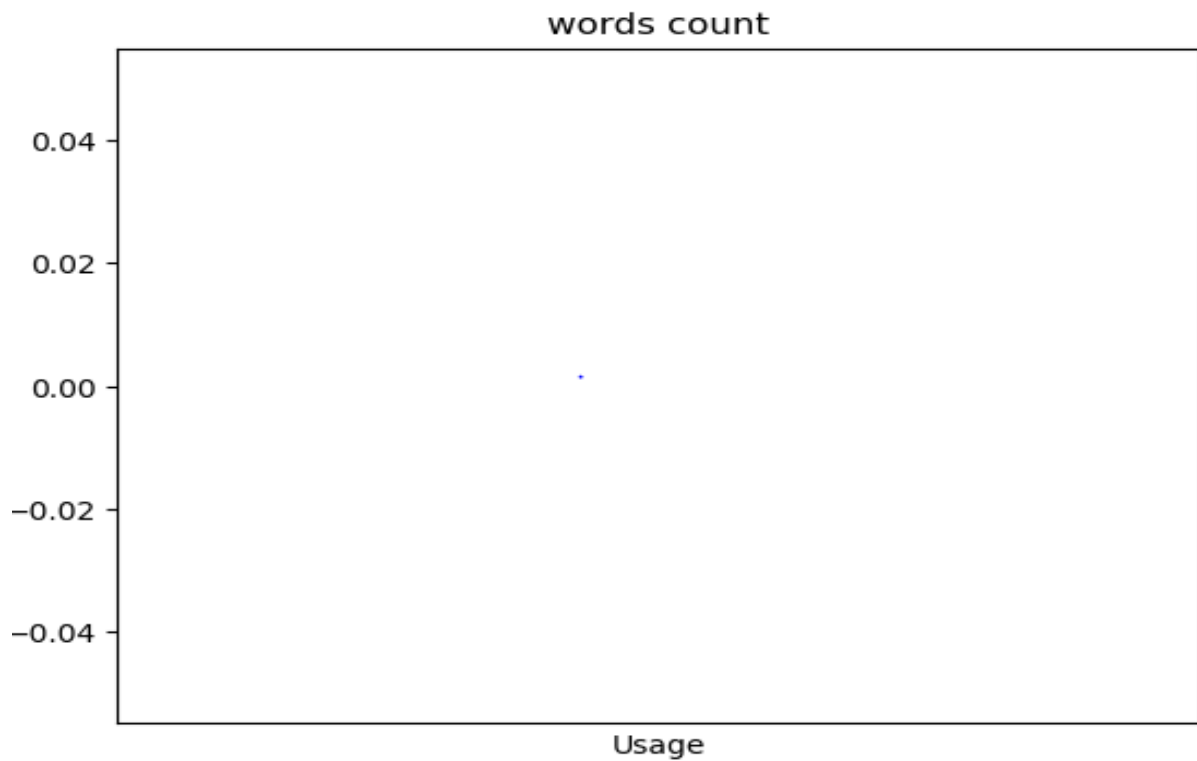


Fig 7.1.c: Negative words in a review

Input2 : Json file(.json ext)

In this case, we are given string of numbers as input so that the output is considered as neutral review because the system does not understand other than english.

```
{
  "reviewerID": "A14R9XMZVJ6INB", "asin": "616719923X", "reviewerName": "amf0001",
  "helpful": [0, 1], "reviewText": "12365 4789 5231 45698 71000", "overall": 3.0, "summary": "3.5
  stars, sadly not as wonderful as I had hoped", "unixReviewTime": 1400457600, "reviewTime":
  "05 19, 2014"
}
```

Output:

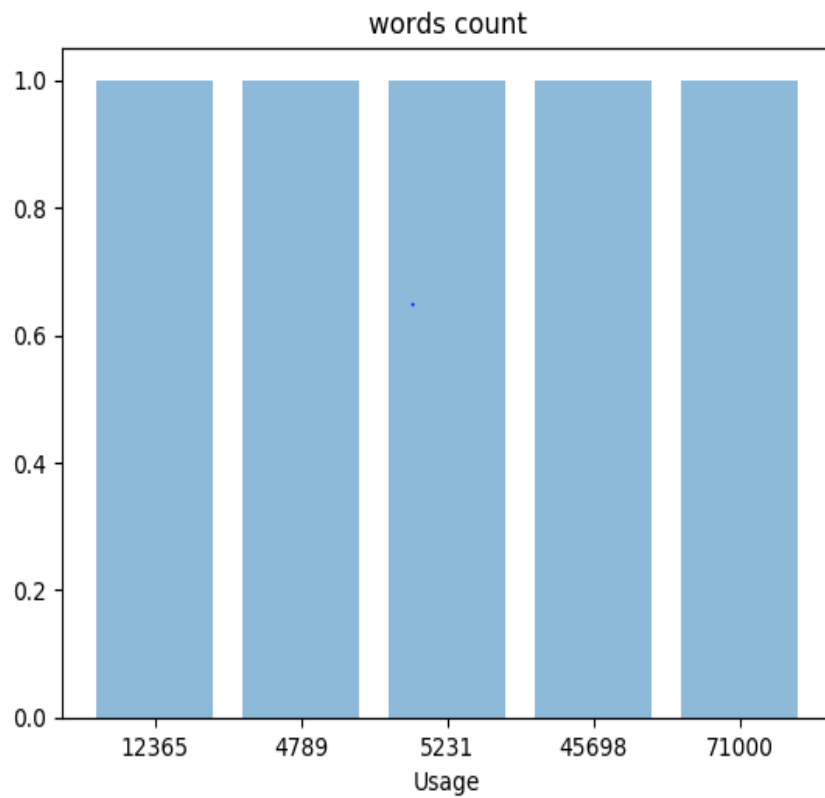


Fig 7.1.d: Neutral words in a review

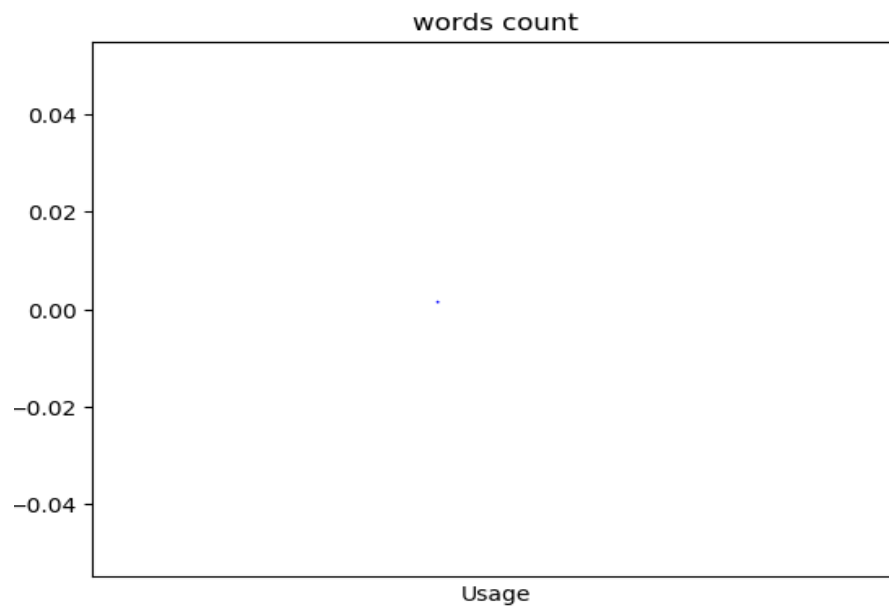


Fig 7.1.e: Positive words in a review

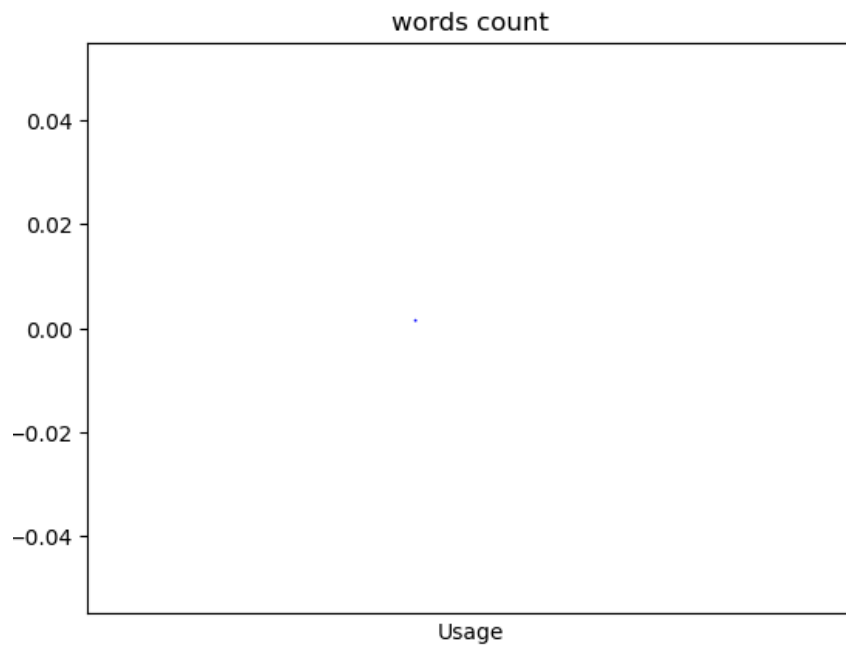


Fig 7.1.f: Negative words in a review

7.2. White Box Testing

White Box Testing is also known as ClearBox Testing, OpenBox Testing, GlassBox Testing, TransparentBox Testing, Code-Based Testing or Structural Testing is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential. White box testing is testing beyond the user interface and into the nitty-gritty of a system.

Fig 7.2.a contains input for original review

```
In [9]: example_sent
Out[9]: 'very good in use bad in batterey'
```

Fig 7.2.a: Original review given by the user.

The following figures represent the stop words that are available in English language

```
In [12]: set(stopwords.words('english'))
Out[12]: {'a',
          'about',
          'above',
          'after',
          'again',
          'against',
          'ain',
          'all',
          'am',
          'an',
          'and',
          'any',
          'are',
          'aren',
          "aren't",
          'as',
          'at',
          'be',
          'because',
          'been',
          'before',
          'being',
          'below',
          'between',
          'both',
          'but',
          'by',
          'can',
```


Fig 7.2.b: Stop words in English language

```
"couldn't",  
'd',  
'did',  
'didn',  
"didn't",  
'do',  
'does',  
'doesn',  
"doesn't",  
'doing',  
'don',  
"don't",  
'down',  
'during',  
'each',  
'few',  
'for',  
'from',  
'further',  
'had',  
'hadn',  
"hadn't",  
'has',  
'hasn',  
"hasn't",  
'have',  
'haven',  
"haven't",  
'having',  
'he',  
'her',  
'here',
```

Fig 7.2.b: Stop words in English language

```
'him',  
'himself',  
'his',  
'how',  
'i',  
'if',  
'in',  
'into',  
'is',  
'isn',  
"isn't",  
'it',  
"it's",  
'its',  
'itself',  
'just',  
'll',  
'm',  
'ma',  
'me',  
'mightn',  
"mightn't",  
'more',  
'most',  
'mustn',  
"mustn't",  
'my',  
'myself',  
'needn',  
"needn't",  
'no',
```

Fig 7.2.b: Stop words in English language



```
'nor',  
'not',  
'now',  
'o',  
'of',  
'off',  
'on',  
'once',  
'only',  
'or',  
'other',  
'our',  
'ours',  
'ourselves',  
'out',  
'over',  
'own',  
're',  
's',  
'same',  
'shan',  
"shan't",  
'she',  
"she's",  
'should',  
"should've",  
'shouldn',  
"shouldn't",  
'so',
```

Fig 7.2.b: Stop words in English language

```
'wasn',  
"wasn't",  
'we',  
'were',  
'weren',  
"weren't",  
'what',  
'when',  
'where',  
'which',  
'while',  
'who',  
'whom',  
'why',  
'will',  
'with',  
'won',  
"won't",  
'wouldn',  
"wouldn't",  
'y',  
'you',  
"you'd",  
"you'll",  
"you're",  
"you've",  
'your',  
'yours',  
'yourself',  
'yourselves' }
```

Fig 7.2.b: Stop words in English language

The following output contains filtered sentence after removing the stop words

```
In [19]: filtered_sentence  
Out[19]: ['good', 'use', 'bad', 'batterey']
```

Fig 7.2.c: Filtered sentence after removing stop words

The following output contains positive words in the given review

```
In [26]: pos_word_list  
Out[26]: []
```

Fig 7.2.d: Positive words in the filtered review

The following output contains neutral words in the given review

```
In [27]: neu_word_list  
Out[27]: ['good', 'use', 'batterey']
```

Fig 7.2.e: Neutral words in the filtered review

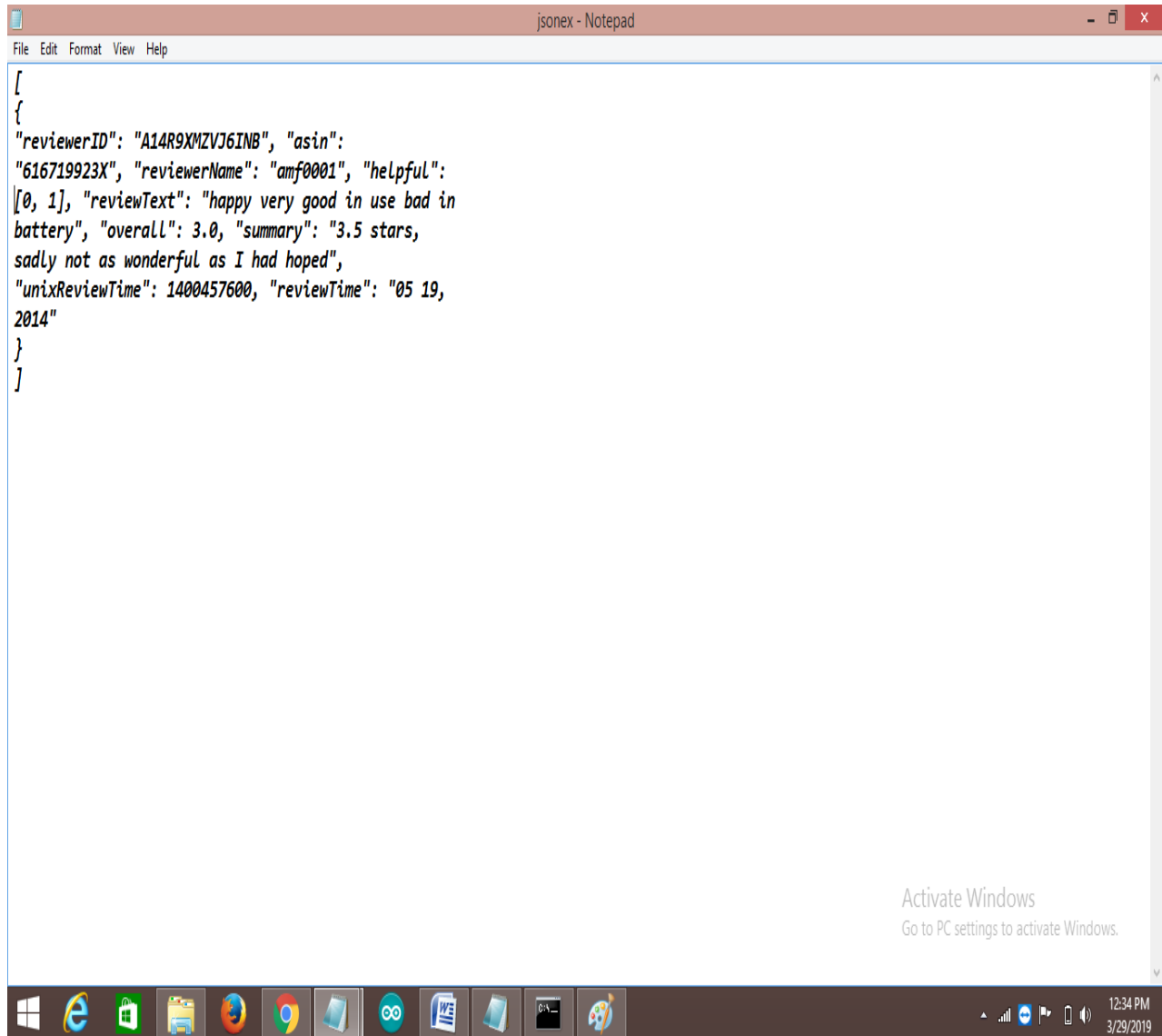
The following output contains negative words in the given review

```
In [28]: neg_word_list  
Out[28]: ['bad']
```

Fig 7.2.f: Negative words in the filtered review

8. Screenshots

Input: Input is in Json format



```
[
{
  "reviewerID": "A14R9XMZVJ6INB", "asin":
  "616719923X", "reviewID": "amf0001", "helpful":
  [[0, 1], "reviewText": "happy very good in use bad in
  battery", "overall": 3.0, "summary": "3.5 stars,
  sadly not as wonderful as I had hoped",
  "unixReviewTime": 1400457600, "reviewTime": "05 19,
  2014"
}
]
```

Fig 8.1.a: Input file.

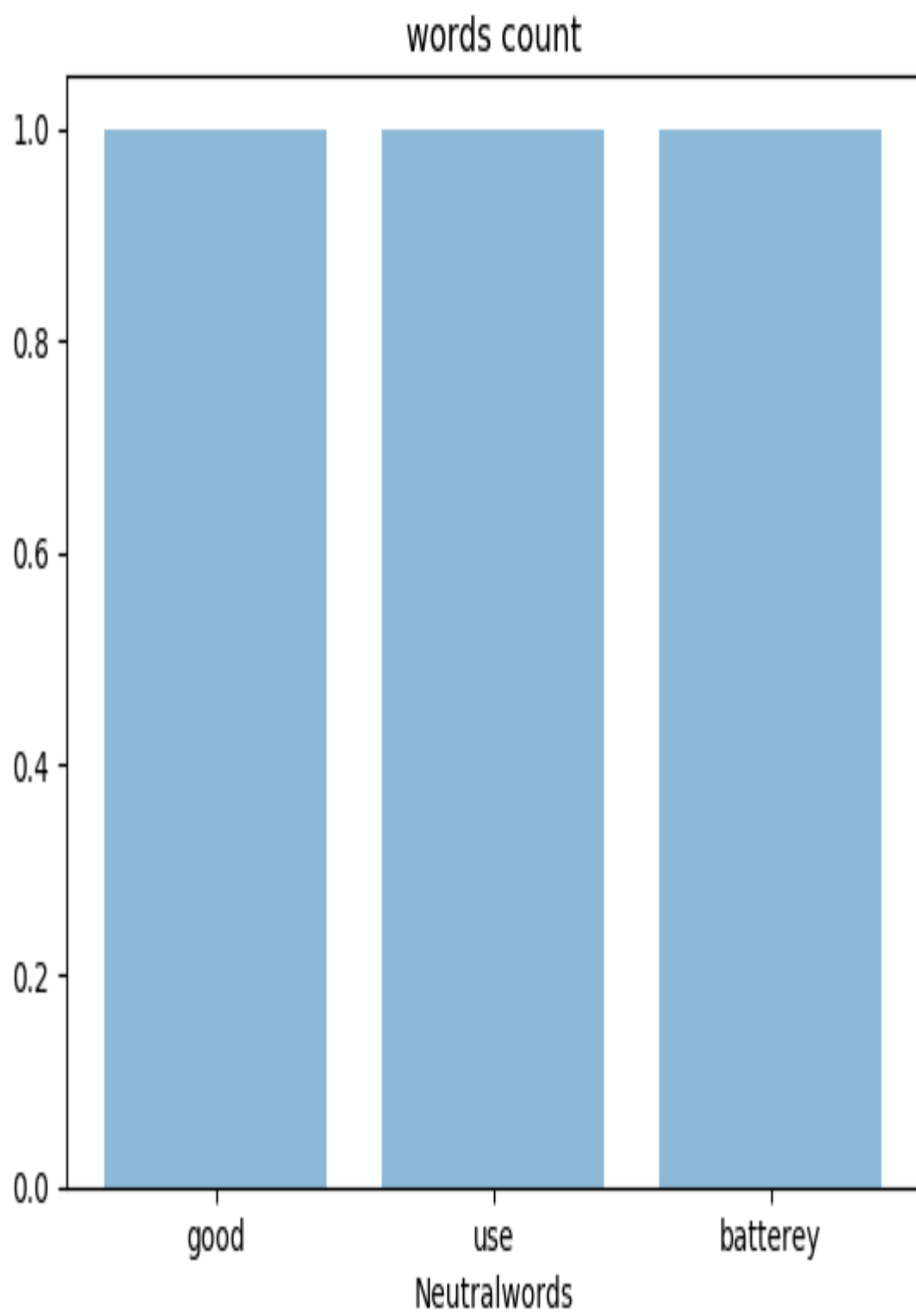


Fig 8.1.b: Graph represents neutral words in a review.

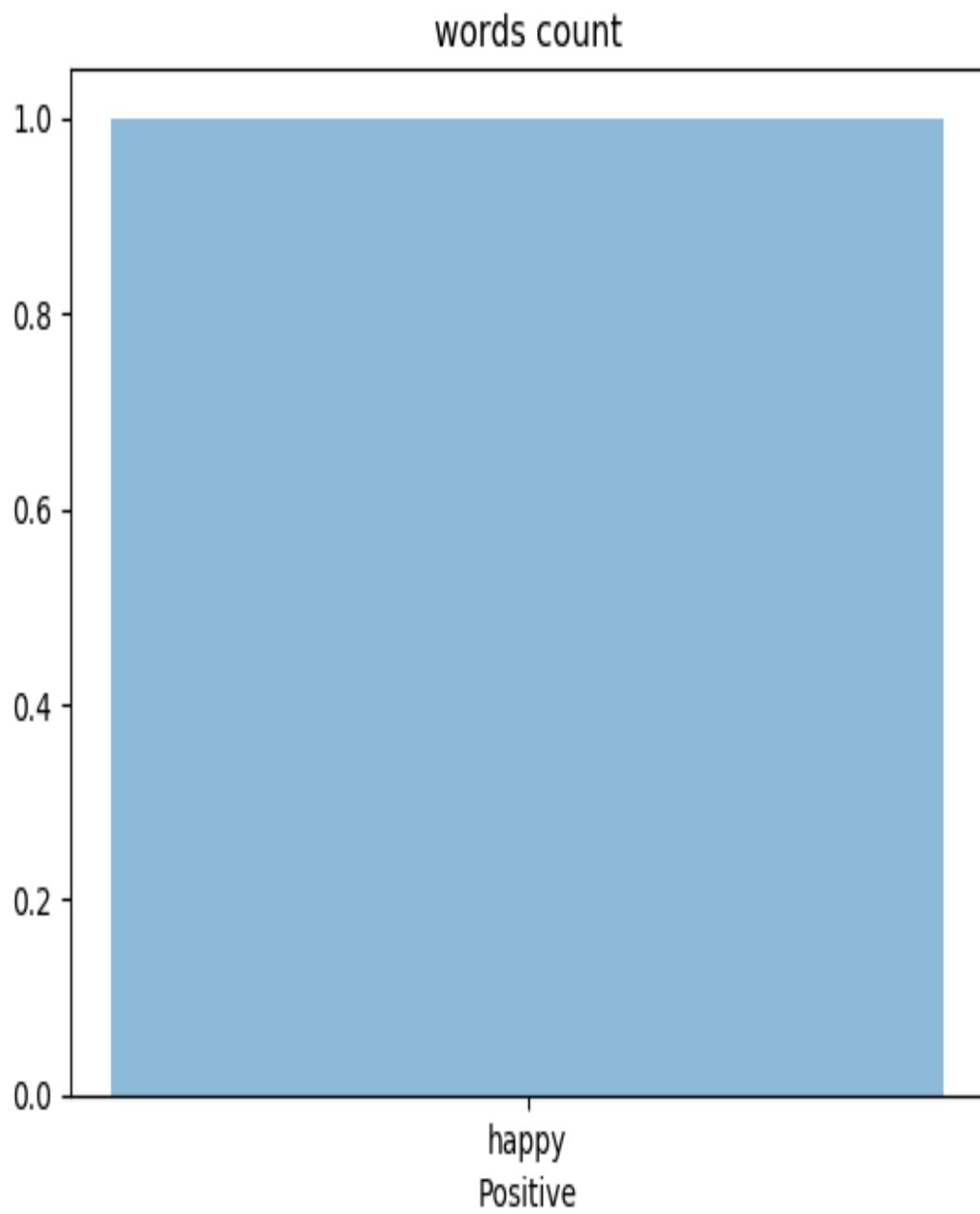


Fig 8.1.c: Graph represents positive words in a review.

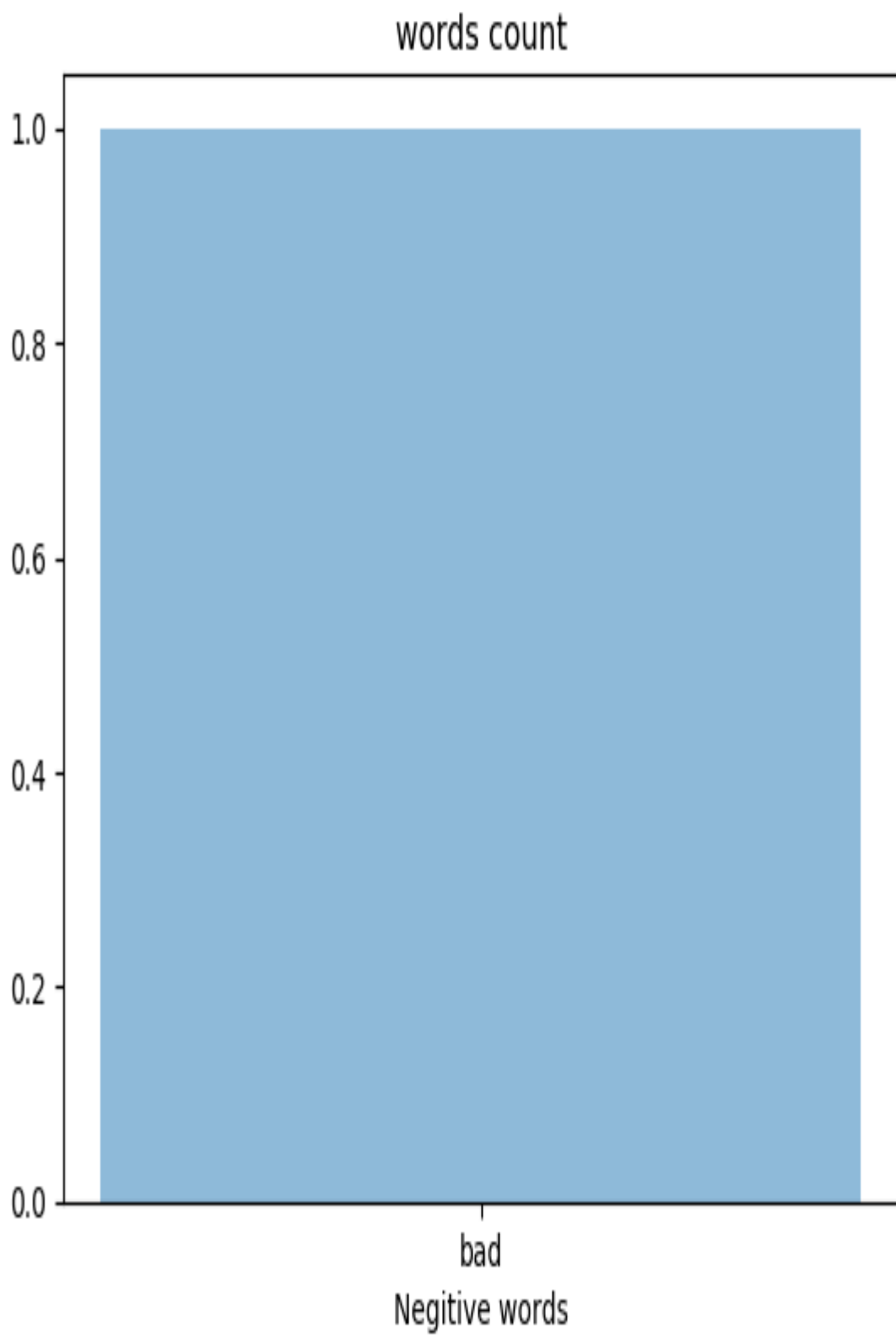


Fig 8.1.d: graph represents negative words in a review.

9. Conclusion and Future Work

In this work, we determined ratings based on reviews by using Natural Language Processing techniques. This helps to predict better ratings and efficient than recommended system algorithms, it also generate recommendations in quick manner and doesn't required any training datasets.

In future, the proposed work may extend by using NLP with K-Means, Decision Tree Algorithms which helps to perform more effective and produce efficient results.

10. Bibliography

- [1] V. W. Zheng, B. Cao, Y. Zheng, X. Xie, and Q. Yang, Collaborative filtering meets mobile recommendation: a user-centered approach, in Proceedings of the 24th AAAI Conference on Artificial Intelligence, pp. 236–241, 2010.
- [2] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang, Towards mobile intelligence: learning from GPS history data for collaborative recommendation, Artificial Intelligence, vol. 184/185, pp. 17–37, 2012.
- [3] C. Ono, M. Kurokawa, Y. Motomura, and H. Asoh, A contextaware movie preference model using a bayesian network for recommendation and promotion, in Proceedings of 11th International Conference on User Modeling, pp. 247–257, 2007.
- [4] M. Szomszor, C. Cattuto, H. Alani et al., Folksonomies, the semantic web, and movie recommendation, in Proceedings of 4th European Semantic Web Conference, pp. 1–14, 2007.
- [5] R. Armstrong, D. Freitag, T. Joachims, T. Mitchell, et al., WebWatcher: a learning apprentice for the world wide web, AAAI Spring symposium on Information gathering from Heterogeneous, distributed environments, (1995), pp. 6–12.
- [6] L. Terveen, W. Hill, B. Amento, D. McDonald, J. Creter, PHOAKS: a system for sharing recommendations, Commun. ACM 40 (3) (1997) 59–62.
- [7] H. Kautz, B. Selman, M. Shah, Referral web: combining social networks and collaborative filtering, Commun. ACM 40 (3) (1997) 63–65.
- [8] T. Joerding, A temporary user modeling approach for adaptive shopping on the web, Proceedings of Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, Toronto and Banff, Canada. Computer Science Report, (1999), pp. 99–107.
- [9] B. Moore, R. Janker, B. Papez, L. Power, R. Watkins, Migrating weblogic applications to websphere advanced edition, IBM Redbooks, (2001), p. 1.
- [10] J. L. Herlocker, J. Konstan, A. Borchers, and J. Riedl, An algorithmic framework for performing collaborative filtering, in Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99), pp. 230–237, Berkeley, Calif, USA, August 1999.
- [11] Y. Koren, Collaborative filtering with temporal dynamics, Communications of the ACM, vol. 53, no. 4, pp. 89–97, 2010.
- [12] T. Hofmann, Collaborative filtering via gaussian probabilistic latent semantic analysis, in

Proceedings of the 26th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pp. 259–266, Toronto, Canada, 2003.

[13] Y. Zhang, D. Zhang, M. M. Hassan, A. Alamri, and L. Peng, CADRE: Cloud-Assisted Drug recommendation Service for Online Pharmacies, *Mobile Networks and Applications*, vol. 20, no. 3, pp. 348–355, 2015.

[14] B. Marlin, Modeling user rating profiles for collaborative filtering, in *Proceedings of the 16th International Conference on Neural Information Processing Systems*, pp. 627–634, 2003.

[15] R. Salakhutdinov, A. Mnih, and G. Hinton, Restricted Boltzmann machines for collaborative filtering, in *Proceedings of the 24th International Conference on Machine learning (ICML '07)*, vol. 227, pp. 791–798, Corvallis, Oregon, June 2007.

[16] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*, pp. 426–434, New York, NY, USA, August 2008.

[17] S. Rendle, Factorization machines, in *Proceedings of the 10th IEEE International Conference on Data Mining, ICDM 2010*, pp. 995–1000, Australia, December 2010.

[18] Y. Zhen, W.-J. Li, and D.-Y. Yeung, TagiCoFi: Tag informed collaborative filtering, in *Proceedings of the 3rd ACM Conference on Recommender Systems, RecSys'09*, pp. 69–76, USA, October 2009.

[19] G. Lekakos and P. Caravelas, A hybrid approach for movie recommendation, *Multimedia Tools and Applications*, vol. 36, no. 1-2, pp. 55–70, 2008.

[20] Y. Zhang, Z. Tu, and Q. Wang, TempoRec: Temporal-Topic Based Recommender for Social Network Services, *Mobile Networks and Applications*, vol. 22, pp. 1182–1191, 2017.

[21] S. Debnath, N. Ganguly, and P. Mitra, Feature weighting in content based recommendation system using social network analysis, in *Proceedings of the 17th International Conference on World Wide Web (WWW '08)*, pp. 1041-1042, Beijing, China, April 2008.

[22] M. Nazim Uddin, J. Shrestha, and G.-S. Jo, Enhanced content based filtering using diverse collaborative prediction for movie recommendation, in *Proceedings of the 2009 1st Asian Conference on Intelligent Information and Database Systems, ACIIDS 2009*, pp. 132–137, Viet Nam, April 2009.

[23] K. Soni, R. Goyal, B. Vadera, and S. More, A Tree Way Hybrid Movie Recommendation System, *International Journal of Computer Applications*, vol. 160, no. 9, pp. 29–32, 2017.

[24] G. Ling, M. R. Lyu, and I. King, Ratings meet reviews, a combined approach to recommend,

in Proceedings of the 8th ACM Conference on Recommender Systems, RecSys 2014, pp. 105–112, USA, October 2014.

[25] Y. Zhang, M. Chen, D. Huang, D. Wu, and Y. Li, IDoctor: personalized and professionalized medical recommendations based on hybrid matrix factorization, *Future Generation Computer Systems*, vol. 66, pp. 30–35, 2017.

[26] P. D. Turney, Tumbs up or thumbs down? in Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 417–424, Philadelphia, Pennsylvania, July 2002.

[27] J. R. Priester and R. E. Petty, The Gradual Threshold Model of Ambivalence: Relating the Positive and Negative Bases of Attitudes to Subjective Ambivalence, *Journal of Personality and Social Psychology*, vol. 71, no. 3, pp. 431–449, 1996.

[28] H. Li, J. Cui, B. Shen, and J. Ma, An intelligent movie recommendation system through group-level sentiment analysis in microblogs, *Neurocomputing*, vol. 210, pp. 164–173, 2016.

[29] J. Sun, G. Wang, X. Cheng, and Y. Fu, Mining affective text to improve social media item recommendation, *Information Processing & Management*, vol. 51, no. 4, pp. 444–457, 2015.

[30] Q. Diao, M. Qiu, C.-Y. Wu, A. J. Smola, J. Jiang, and C. Wang, Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS), in Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14), pp. 193–202, ACM, New York, NY, USA, August 2014.

[31] Z.-D. Zhao and M.-S. Shang, User-based collaborative-filtering recommendation algorithms on hadoop, in Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, WKDD 2010, pp. 478–481, Thailand, January 2010.

[32] Sweden's largest Facebook study: a survey of 1000 Swedish Facebook users - University of Gothenburg, Sweden. [En ligne]. Disponible sur: <http://www.gri.handels.gu.se/english/latest-news/news/d/sweden-s-largest-facebook-study--a-survey-of-1000-swedish-facebook-users.cid1073014>. [Consulte le: 07-nov-2012].

[33] Rana Chamsi Abu Quba On enhancing recommender systems by utilizing general social networks combined with users goals and contextual awareness HAL Id: tel-01236089 <https://tel.archives-ouvertes.fr/tel-01236089>.

