FACULTY OF ENGINEERING AND TECHNOLOGY

MASTER OF SOFTWARE ENGINEERING

MASTER THESIS

Automatic Classification of Apps Reviews for Requirement Engineering (Exploring The Customer's Need from The Healthcare Applications)

Author:
Nadeem AlKilani (1155073)

Supervisor:
Dr. Abualsoud Hanani

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in Software Engineering at Birzeit University, Palestine

February 12, 2019

Approved by the thesis committee:

_____

Dr. Abualseoud Hanani, Birzeit University

_____

Dr. Ahmad Afaneh, Birzeit University

_____

Dr. Majdi Mafarja, Birzeit University

_____

Date approved:

_____

Declaration of Authorship

I, Nadeem AlKilani , declare that this thesis titled, **"Automatic Classification of Apps Reviews for Requirement Engineering (Exploring The Customer's Need from The Healthcare Applications) "** and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a master degree at Birzeit University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

Abstract

Millions of mobile applications are available for download at the applications stores. Millions of users write reviews expressing their daily experience on applications they used. these reviews contain valuable information for application vendors and developers. However, due to the amount of data, traditional searching algorithms don't work to extract effective review information. Machine learning techniques are efficiently used to extract significant information for software requirement Engineering. In this project, we intend to apply machine learning techniques to classify health care application reviews into several types such as bug reports, new feature requests, application performance and accuracy and user interface. There is no available free annotated data set for training and evaluating machine learning techniques, therefore, more than 7500 reviews for 10 different health-related mobile applications are annotated manually by experts in the field. Our experiments show that Multi-nominal Naive Bays can classify mobile apps reviews into bugs, new features and sentimental with an accuracy of 87%, and into general bug, usability,security and performance with an accuracy of 88%. The best result of the sentimental analysis system is 90%. in addition, the experiments show that the overall performance is improved when we use the data subset with high confidence labels and when two experts agree on the same label. The Re-sampling technique was successfully used to overcome the data imbalance problem in our data set, the accuracy improved to 89% for mobile apps reviews into a set of classes; bugs, security, new feature, performance, and usability and 96% for the sentimental reviews.

ملخص

مع التطور المستمر لتطبيقات الهواتف الذكية، وتوفر الملايين من التطبيقات على المتاجر، يوجد هناك عدد كبير جدا من التعليقات المفيدة من مستخدمي هذه الاجهزة، الذين يقومون بإبداء آرائهم البنّاءَه من أجل تحسين هذه البرامج في النسخ الجديدة وهذا ما يتيح الفرصة للشركات المبرمجة لهذه التطبيقات لتحسين هذه البرامج والاهتمام بزبائنهم لتلبية حاجاتهم المستمرة والمتجددة.

ولكن كما هو معروف من الصعب جداً القيام بقراءة وتنظيم وتصنيف هذه التعليقات بالطريقة التقليدية وهي الطريقة اليدوية حيث انها تحتاج وقتاً كبيراً جداً وتحتاج ايضاً الى مبلغ ضخم من المالوباستخدام خوارزميات تقليدية تفتقر للجوده.

ومع تطور علم البيانات والتعامل مع البيانات الضخمة التي تحوي كماً كبيراً جداً من المعلومات، قمنا في هذه الرسالة بتطبيق آليات حديثة للتعامل مع آراء مستخدمي هذه التطبيقات الصحية للهواتف الذكية ، وذلك للعمل على تصنيف هذه البيانات الى خمسةِ أقسامٍ رئيسيةٍ كالتالي : إذا كان المشترك يعطي تقريراً عن مشكلة في البرنامج او يريد اضافة ميزة جديدة أو بطؤ في البرنامج او ان المشترك يشكو من دقة بيانات البرنامج او وجود ثغرة امنية في البرنامج ، وفي نفس الوقت قمنا بتصنيف شعور المستخدم حول التطبيق الصحي ماإذا كان راضٍ عنه أو غير راضٍ أو بين ذلك.

واجهنا تحدي كبير جداً بجمع قرابة ال ٧٥٠٠ تعليق حقيقي من خلال ١٠ تطبيقات صحية للهواتف الذكية وقمنا من خلال خبراء فنيين بعمل تصنيف يدوي لهذه التعليقات بحيث تم تعليم الخوارزميات المستخدمة لعمل نمط معين من خلاله يتم تصنيف البيانات الجديدة بشكل تلقائي.

وبعد القيام بعدة تجارب مختلفة من توليفات دقيقة حصلنا على دقة تصنيف تصل d الى ٨٧ % في خوازمية Multi-nominal Naïve Bays

بالنسبة لشعور الاشخاص حول التطبيق المستخدم من قبلهم و ٨٨ % في المجموعة الثانية من التصنيفات التي تهتم بتصنيف متطلبات المستخدم.

ومع التحسين المستمر للبيانات واستخدام الخوارزمية المناسبة و مع التوليف المناسب حصلنا على نتيجة ٩٠ % من دقة تصنيف التعليقات الجديدة وبشكل تلقائي. من الجدير بالذكر، أن دقة التصنيف كانت أكبر كلما كانت جودة البيانات أكبر، ومدققة من أكثر من خبير في نفس الوقت. أيضا عندما تم إلغاء التحيز لمجموعة معينة من البيانات وجعل جميع المجموعات من البيانات متساوية، حصلنا على نتيجة ٩٦ % من الدقة في شعور المستخدمين و ٨٩ % في المجموعة الثانية من التصنيفات التي تهتم بتصنيف متطلبات المستخدم.

Acknowledgements

Praise is to Allah, the Almighty for guiding me at every stage of my life. I would like to thank the following people, without whose help and support this thesis would not have been possible. First I like to thank my supervisor Dr. Abudalsoud Hanani for his suggestions, encouragements, guidance in writing this thesis and approaching challenges. I am one of those fortunate students to scribe my name in his student's list.

Moreover, I would like to thank my parents, my wife and my friends for their constant support during the time I studied and their help in the data set annotations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Requirement elicitation techniques are very important for helping stake- holders interact with software Requirement Engineers (RE), in addition it helps RE understand the problem domain risen from stakeholders. These techniques could play a key role in resolving conflicts that may result from inconsistency between requirements, which require clear and concise questions from RE and stakeholders [36]. It will also help the RE appropriately determine stakeholder's needs and what solutions well suit their requirements.

The interesting part of the requirement elicitation is that it will save time and money, in addition, it will make requirements more clear. The success of the requirements elicitation gives an indication of the success of the project, since it is considered as the core phase of the whole development life cycle of software production. According to the Standish Group CHAOS report [36], approximately, 50% of projects were scrapped due to the failure of good requirements elicitation.

Huge amounts of mobile application reviews created by users can be utilized for marketing purposes, it can assist in measuring customer satisfaction. Reviews can also be useful in gathering user needs and better understanding the customer base. In this work, nonfunctional requirements are automatically extracted from the mobile apps reviews. Machine learning techniques are used to do this purpose by classifying each user review into one of a predefined classes related to the software requirement.

The interest in healthcare related apps is growing significantly. Health- care systems in many countries rely on mobile applications which provide essential services for patients. For this reason, and because of the diversity of mobile apps, the experimental work presented in this thesis is applied to the user reviews of healthcare apps.

User reviews of mobile applications often contain the users experience of using software products in the form of complaints and suggestions, as well. Such information can be utilized by software developers in order to early fix bugs and enhance the new release of the software systems [26]. However, due to the noisy-nature of those reviews and large volume of them , the manual analysis of them is really a big challenge to get a useful information.

Mobile applications are being used by billions of users across the world, thousands of reviews are posted on the app stores every day. These reviews can be very valuable for developers and application owners to gather requirements, feedback, and recomendation directly from users. Yet, reading this huge number of reviews manually, or using traditional means can be definitely time-consuming

and cumbersome. Furthermore, user's reviews are unstructured and often written unprofessionally, and unclear. Analyzing these reviews can provide many insights to enhance, improve applications and user experience, and more importantly respond to users feedback to increase user satisfaction.

Nowadays mobile applications play an important role in people's life. These applications provide services in different domains for different social groups. Users seeking healthcare applications is increasing exponentially. With the revolution of the information and communication technologies, users are increasingly using the internet to access health services where they can find information that help them take care of their health.

With such a technological revolution, patients can keep in touch with a doctor from their home [28]. Doctors can do some basic diagnosis for their patients, in. Telemedicine and telehealth applications have been introduced in order to make healthcare more convenient, less expensive, and more preventative. Healthcare mobile applications offer interesting opportunities in the healthcare industry, by expanding services to patients beyond clinic boundaries. Health applications dominate "app store"; therefore, it can be viewed challenging to developers to gather user's needs and build mobile health applications that have special features concerning the well being of patients. In addition, such applications require high information security measures in order to ensure privacy.

Mobile health applications should be characterized by accuracy, high performance, security, privacy and many other aspects. Hence, any faults or delay could cause a negative impact on patients' health and data privacy resulting in loss of confidence.

The main aim of this work is to apply machine learning and data mining techniques to extract valuable information for healthcare software requirement engineering from a large quantity of user reviews.

1.2 Background

In the last decade, we have seen a large growth in the usage of smartphone technologies in the health industry. Doctor's interaction with patients and vice versa plays the most vital role in healthcare delivery. In fact, it is a significant challenge to monitor the patients remotely. However, in this advanced technology age, it is not difficult to bridge the gap between a doctor and a patient.

The healthcare industry is excited for smart- phone applications that connect with patients anytime, anywhere, and improves the operating performance of their clinics. Moreover, people are seeking quick and easy solutions for their health ailments.

Smartphone applications help healthcare organizations to deliver good quality care services[38],improved workflow processes as well as increased patient's interaction. Typically, this minimizes the complexity, reduces cost, and time in order to achieve the desired goals. The adopting of smart- phone technology and new conventional approaches to healthcare services have increased the demand for smartphone health applications.Smartphone healthcare applications make you provide the best treatment for optimum care, anywhere and anytime. It offers health- care tools that enhance treatment quality and streamline workloads.

Healthcare providers[17] monitor the patient's condition and respond proactively before it leads to any chronic disorder. They assist follow up patient's health records, updating health plans and maintaining the communication and collaboration with the patients.

Physicians[38] use smartphone health applications in storing patient medical information without any problems. It simplifies the complexity for healthcare providers in making decisions on the patients' diagnosis through patient medical history records. Obviously, a well- designed user interface application can help physicians to efficiently track the specific patient-related information.

Patients can book an appointment with physicians through smartphone applications, It helps healthcare providers to organize their availability and notifies them when a patient books or cancels a medical-appointment.

Cutting-edge Smartphone technologies have led to many great mobile health applications in application stores, such as Android and iOS, which are the two largest platforms that host more than 165,000 applications regarding the medical sector. However, 9% of it addresses topics such as screening, diagnose and monitoring widespread spectrum illnesses [1]. Smartphone technology innovation outpaced the critical evaluation of the impact of smartphones' medical applications. Moreover, medical experts criticize the current situation of smartphone medical applications, because these applications are predominantly technology-driven and thus fail to meet the requirements of the clinical practices [23].

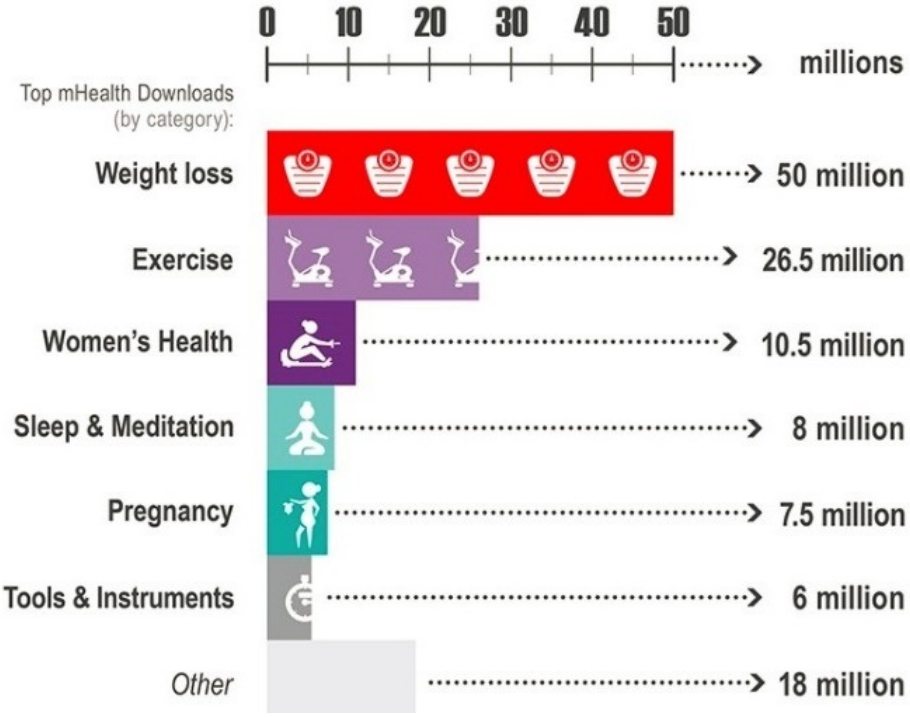

FIGURE 1.1: Percentage of the download of the healthcare application grouped by category. [11]

The low involvement rate of medical professionals does not only increase the risk that ineffective or even the potential harmful tools that will be used by the patients; it also leads to a poor rate of acceptance of the smartphone medical applications among physicians and therefore a low integration of these tools into the daily clinical practices [35].

Smartphones medical applications leverage modern smart devices such as smartphones and tablets with built-in sensor technology, related software development kits (SDKs) to screen, diagnose, and monitor the patient's illnesses. Screening is the routine examination of individuals indicating illnesses or high-risk illnesses. Diagnosis is the inferred state that indicates an illness is present in the patient. The smartphones patient monitoring uses "technology to manage,

monitor, and treat patient's illness from a distance" [21] once an illness has been attributed to a patient. There are a number of studies focused on the development of smartphone application solutions for specific illnesses, such as diabetes [3], asthma, and depression, and have reported lessons learned. Goyal et al. [13] took a user-centered design approach, ensuring that the features of smartphone applications are informed by the needs of patients with Type 2 Diabetes. The resulting application allows the patients to self-monitor their physical activities, diets, and weights, to identify glycemic control patterns in relation to their lifestyles, and to guide them towards remedial decision-making. Årsand et al. [3] illustrates that their smartphone applications can motivate Type 2 Diabetes patients to think about how they can improve their health. Another recent study presents a remote monitoring system for elderly patients with multiple chronic conditions [4], which allow users to see their current medical reports on their smartphones based on sensor data, to perform new measurements, and to communicate with care givers via the smartphone applications.

There has been a lot research on the reliability of Medical smartphone applications that are available on app stores that shows how many of these application lack empirical evidence and short comings which has some serious consequences in result. For instance, Wolf et al.[5] measures the performance of four smartphone applications that evaluate photographs of skin lesions, when such a picture is evaluated, the smartphone application gives the user a feedback about the likelihood of malignancy. The sensitivity of the investigated smartphone applications ranges from 6.8% to 98.1%.

Another example where Hamilton and Brady[24] analyzed up to 111 medical smartphone applications focusing on pain management, link the weak performance of some existing smartphone applications to low professional medical involvement in the design of these applications, other studies [35] found that the content of smartphone applications contains misleading claims and a lack of academic references.

Machine learning algorithms are pretty helpful to classify and predict whether a document represents positive or negative sentiment. Machine learning falls into two categories: supervised and unsupervised algorithms. The supervised algorithm uses a labeled dataset where each training document is labeled with an appropriate sentiment. Whereas, unsupervised learning includes unlabeled dataset where the text is not labeled with appropriate sentiments.

Supervised learning is fairly common in problem classification because the goal is to make the computer know the created classification system. In the literature, there are many different classifiers based on supervised learning algorithms, such as Naive Bayes (NB), Support Vector Machines (SVM), Maximum Entropy (MaxEnt), Binary Classifiers (BC), Decision Tree Classifiers (DTC), Random Forest Classifiers (RFC), etc. However, NB, SVM, and MaxEnt are most commonly used algorithms in the field. Pang and Lee [32] have labeled sentences in the document as subjective or objective . They have applied machine learning classifier to the subjective group, which prevents polarity classification from considering any misleading data. They have explored extraction of methods on the basis of minimum-cut formulation, which provides an effective way for the integration of inter-sentence level information with many words [37].

Opinion mining and sentiment analysis involve the extraction of sentiment words from user reviews, automatic classification, and summarizing of sentiments.

Sentiment words in the free text can be identified by considering the following: adjectives or adverbs, uni-grams or n-grams with their frequency of occurrence, the POS (parts of speech) tagging of words or the negation of words [9].

1.3 Objectives

The main aim of this work is to apply machine learning and data mining techniques, as well as, some of Natural Language Processing (NLP) techniques to extract non-functional software requirements from user's review of a selected group of healthcare mobile applications. This information can be then used for updating and improving mobile application accordingly. The main objectives of this work can be summarized as follow:

- Collect sufficient amount of user's reviews for mobile apps in the healthcare domain from different apps stores.
- Use the annotated dataset for developing an automatic system, which can classify unseen user review into one of the target classes, using machine learning and data science techniques.
- Evaluate the developed systems and study the effect of different machine learning techniques and data quality and quantity.
- Make all collected and annotated dataset and developed systems available for researchers who like to do work in this field.

1.4 Research Questions

Although there is some work has been done in this field and related field, we believe that still there are some interesting research questions that this work can contribute to finding answers to these questions. Some of these questions can be formulated in the following three questions:

RQ1: To what extent user' reviews can be useful and beneficial for software requirements engineering?

RQ2: How effectively can the machine learning and data mining techniques are used to classify unstructured users posts/reviews into non-functional requirements?

RQ3: What are the most useful features in the natural language processing and the most efficient machine learning technique for this task?

The important chapters in the thesis that answer our question are **Chpater4 Data Collecting and Analysis** which is dedicated to full-fill our **first objective**, and shed light on answering the **RQ1**, and **Chapter5 Experiments and Results** which is dedicated to full-fill our **second and third objectives** and shed light on answering the to the **RQ2** and **RQ3**.

Chapter 2

Literature Review

In this section, a detailed review of related and recent work is presented. To make the literature reviews more organized, we divide the literature review into sub-sections, starting by presenting a review of related studies in the field of sentimental analysis, which is related and close to the scope of this study. After that we present a detailed review of similar work to our work, focusing on the data set used, features representation and machine learning techniques used in these studies.

2.1 Sentimental Analysis

The customer's satisfaction predicts their retention, loyalty, and products repurchase. It has also been suggested to get the impact on the future's product search activity, alterations in "hopping behavior", as well as the trials of other available products in this sector. Keiningham and Vavra [22] found that for every increased percentage of the customer's satisfaction, there is an increased average of 2.37% that returns on the investment. Furthermore, when customers are satisfied, they spread this information, and act as the company marketers. According to Gitomer [12], nearly one half of the American business is built on

"word-of-mouth" communication. Such outcomes demonstrate the importance of customer's satisfaction. While some of these ideas and outcomes are disputed, the measurement of customer reactions remains important. Thus keys are able to assess these reactions, in order to determine methods to improve business effectiveness. The measurement of customer's service is highly important, there is no approved universally measurement scale. According to [29], there are several theories that propose how we should approach the customer's service assessment.

Today, social media represent the networks support that allows the consumers to tap into a vast universe of same opinions, critiques, feedback, recommendations, or warnings about specific products or services that enable the customer to review your site and provide a positive effect on your company's organic search rankings in search engines.

The number of information posted on social media is in the text form. Such texts are not only containing factual information but also it contains subjective evaluations and expressing opinions. These consumer's opinions are of particular value for tourism and hospitality marketers, which aims at gathering the market intelligence, and also for consumers who need to gain a quick overview of the collective opinions about a destination, provider or product/service/experience. Due to the sheer amount of the available consumer-generated content, manually extracting sentiments have become impractical, spurring a whole field of the commonly referred inquiry to an analysis opinion.

Sentiment analysis aims at providing the judgment of the expressed opinion in a subjective text, which is capable of distinguishing positive, negative, or even

a more subtle opinion, such as anger, grief or joy. This involves two subsequent tasks: identification of subjective/objective information, and the sentiment classification of the subjective information, which involves both polarity and strength.

Sentiment analysis has been in practice on a variety of topics. For instance, the sentiment analysis studies for movie reviews, product reviews, and news and blogs. In this context, Twitter-specific sentiment analysis approaches are reported. The research on sentiment analysis so far has mainly focused on two aspects: identifying whether a given textual entity is subjective or objective and identifying the polarity of subjective texts [34]. Most sentiment analysis studies use machine-learning approaches. In the sentiment analysis domain, the texts belong to either positive or negative classes. They may also be a multi-valued or binary class like positive, negative and neutral (or irrelevant).

2.1.1 Sentiment Analysis Levels

According to [27], there are three main classification levels:

- Document level, which classifies a document opinion such as expressing a positive or negative opinion or sentiment whole document as a unit .
- Sentence-level, which classifies an expressed sentiment in each sentence. If the sentence is subjective, then it classifies it in negative or positive opinions.
- Aspect-level, which classifies the sentiment with respect to the specific aspects of entities. Users can give different opinions on different aspects of the same entity.

According to [33], the document sentiment classification approach is typically used to classify movie reviews by means of using supervised machine learning method. In [31], the authors used the semantic orientation of words defined by and several information from the Web and thesaurus. They achieved 85% of accuracy with and the semantic orientation of words and the lemmatized word unigram.

2.2 Extracting Non-Functional Software Requirements Engineering Features from Mobile App Reviews

Maalej and Hadeer [25] tried to classify the user's reviews into bugs, new feature, the user experienced and rated categories. They used different techniques of data classifications and then compared their accuracies.

They applied different techniques as borrowed from the Natural Language Processing, sentimental analysis and text classification, such as Naive Bayes, Decision trees and Maximum-Entropy (MaxEnt).

In order to make their study, they crawled a real users' review from Apple and Google store. The data contains user text review, title, app name, category, store, submission date, username, and star rating.

The main result of a set of experiments conducted, they compared the accuracy of simple string matching, text classification, natural language processing, sentiment analysis, and review metadata to classify the reviews.

They conclude that text classification should be enhanced with metadata such as the tense of the text, the star rating, the sentiment score and the length.

Moreover, stop word removal and lemmatization, which are popular NLP techniques used for document classification, should be used carefully since every word in a short informal review can be informative. Overall, the precision and recall of the four target classes are encouraging –ranging from 71% up to 97%.

Claudia Jacob and Rachel Harrison [18], built a model that only concerns with feature request on the mobile app reviews that were written in the English language. They depend on the linguistic rules and they defined the most words that comprise about 80% of the feature sentences, such as: add, allow, complaint, could, hope, if only, improvement, ... etc. Therefore, they chose multi-categories of the Mobile Applications, such as health, sports, education, ... etc. In order to get a significant sample of the reviews that contain feature request, they used a total of 161 apps and collected a total of 3279 reviews.
They labeled the feature reviews manually; then used the MARA tool, which is designed by them (Mobile App Review Analyzer) [18] to retrieve all the reviews available for each app and classify the feature request.

In the last couple of years, many researchers have suggested probabilistic approaches which can summarize informative review content. Guzman and Maalej [15] applied Natural Language Processing and sentiment analysis in order to extract software features from the user reviews together with a summary of the user opinions about each feature. These information provides valuable assistance to analysts along with quantifying the importance of the software features and prioritizing their work for future releases. As a final conclusion and summary, text classification is highly recommended and should be enhanced with metadata such as the tense of the text, the star rating, the sentiment score and the length. In addition, stop word removal and lemmatization, that are popular

NLP techniques used for document classification, is to be used carefully, since every word in a short informal review can be informative.
They used topic-modeling techniques to group fine-grained features into more meaningful high-level features. They evaluated their approach with 7 apps from the Apple App Store and Google Play Store and compared their results with a manual, peer-conducted analysis of the reviews.

They also extracted the features from the reviews by applying a collocation finding algorithm and aggregating features by their meaning, after they removed the preprocessing text and the noise text.

They get a precision up to 91% (59% average) and a recall up to 73% (51% average). Overall, the results showed that the generated summaries are coherent and contain the most mentioned features in the reviews.

Hui Yang and Peng Liang [40] proposed an approach that aims to extract the functional and non –functional requirement from the mobile application user's review. Their experiment only used iBooks English applications. They also studied the effective number of the manual classification reviews for both functional and non-functional on the F-measure, recall, and precision of the result.
In order to extract the requirement, they used a combination of TF-IDF and NLP techniques, with the manual classification of the review in functional and non –functional requirement. They also removed the stop words and the spam reviews to enhance the accuracy.

They noticed that the value of F-measure dramatically increases when the sample size initially increases from 1 to 20 for FR classification, and from 1 to 7 for

NFR classification. After that size (number) the value of F-measure tends to be stable, this concludes that NFR classification requires fewer sample reviews to get a decent set of keywords to reach a stable F-measure than FR classification. This is reasonable because the FR keywords are more domain-dependent than the NFR keywords. The result of the F-measure for both functional and non-functional requirements for the maximum sample size that were taken was 100, which was the NFR classification results (0.825) more than the FR results (0.479).

Suhaib Mujahid et al.[30] made a study on the wearable app reviews to explore the complaints of the customer and categories. These complaints help developers to focus on its categories and enhance the new release of the application to avoid these problems on new wearable applications; those applications that run over the wearable devices like watches, fitness trackers, etc.

The authors decided to get the applications that had more than 100 low rated (1 star and 2 stars) reviews. Low rated reviews have more complaints due to the study concern.

The final sample was that they made a study of 589 reviews that were varying from 115-154 reviews per each application. The authors manually classified these 589 reviews into 15 complaints' categories and ranked the categories from 1- 15 based on the frequency of the complaints. The study revealed that the most frequent complaints with relates to the functional errors, namely the high-cost and the lack of functionality.

Chaochang Chiu [7] extracted about 2,000 reviews from game applications in Chinese. So, they execrated these reviews and analyzed them using various types of metrics, such as games' attributes, gender, and game types. They found

that there is a high correlation between game attribute and gender; this will support the developers when they design games to take these opinions into their consideration.

their study also revealed that males tend to write reviews when they extremely dis-satisfied or satisfied, in contrast of females who are taking into their consideration more of what do these reviews mean and their decencies with a star rating. They analyzed the reviews into various metrics, which included game' types, such as racing, sports, puzzle, ... etc., and game' attributes, such as stability, developers, aesthetics, and musicality.

Their study reveals the uniqueness of the gender reviews opinions and their rating to the applications; it captures negative and positive reviews with application attribute. Their study is very important to the commercial market for the current applications' offers and the users' needs.

2.3 literature review summary and Conclusion

As we discussed early in this chapter, various researchers in this field proposed some techniques to classify the user's reviews into bugs, new features, the user experience and rated categories. Also, there are some researches about how to classify the movie and sports. The researchers made a study on the wearable app reviews to explore the complaints of the customer and categories.

Some researchers built a model that only concerns with feature request on the mobile app reviews that were written in the English language. Some researchers tried to extract the functional and non –functional requirements from the mobile

application's user's reviews. Additionally, they extracted these reviews and analyzed them using various types of metrics, such as games' attributes, gender, and game types.

They used various techniques and machine learning algorithms such as as Naive Bayes (NB), Support Vector Machines (SVM), Maximum Entropy (MaxEnt), Binary Classifiers (BC), Decision Tree Classifiers (DTC), Random Forest Classifiers (RFC), etc. However, NB, SVM, and MaxEnt are most commonly used algorithms in the field. Of course, we really thank them about their informative researches that leads us to make a new research. My research paper distinguishes to their research that my study is concerned about heath care app reviews. So I build a new data set, which that I collected about 100,000 reviews and the experts manually classify the reviews to five categories, which are usability problem, performance problem, security problem, bug or new feature, also we asked the expert to classify the user's impression to the positive, negative or neutral.

Also our system filters the data into three sub systems, it classifies the users review to main three categories. Then it deeply digs to explore the kind of the bug to general bug, usability, performance or security bug and the classify the kind of the sentimental to positive negative or neutral. We applied used many algorithms and preprocessing, Bigram, etc.

We will elaborate this in the methodology chapter of this study. Our approach helps app analysts and developers to identify useful feedback form the user's reviews .

Chapter 3

Research Methodology

In order to fulfill the objectives of this thesis and introduce answers to the research questions presented in the introduction of this report, two important things need to be prepared and designed well. The first one is a suitable and sufficient dataset, which includes a sufficient number of mobile applications user reviews related to the field of this study, which is the healthcare mobile applications. In addition to the reviews, we need to have some annotations such as the sentimental analysis or the impression of the user about the used application as a positive impression or negative or neutral. We also need to categorize each review into one of our five specified categories (or classes), which represent non-functional requirements engineering. If they are not available with the dataset, these annotations need to be done manually by experts in the field. More details of the dataset collection, description and analysis are presented in the next chapter (chapter 4).

The second important component of our methodology is to design and implement an automatic system, which can classify any unseen review, in our target field, into one of the specified classes with high accuracy and in short time.

Now, how accurate the system is, depends on many factors; such as quality and quantity of the data set used to train the system, the features used to represent the target classes, machine learning technique used for modelling target classes, and other factors. In this chapter, we propose an overall system, which can be used for extracting useful information (e.g. non-functional software requirements) for the user reviews. This information is helpful for the requirement engineering, which is an important part of the software engineering cycle. In our methodology, we achieve this objective by building a classification system, which is able to classify a given review into one of predefined classes that represent main and important categories useful for the software requirement engineer.

Its worth to mention that our system consist of the three main sub-systems, the first one is front end system classify the reviews in to Bugs, New feature or Sentimental, the second sub system is back end system that is responsible to make more digging to explore the details behind the Bugs , so that back end system classify the Bugs into four categories, which are General Bug, Security problem, usability problem or performance problem , and the last one is he sentimental analysis system , which get specific classification of the sentimental, theses specific classifications are positive, negative or neutral.

In order to make the right comparison between the different criteria of the data selection and the techniques that we used , we start our work with baseline system as a convention, so we used it as a reference system for all proposed systems and experiments.

The following sections describe in details the classification system proposed in this work. This system consists of mainly two parts; feature reduction module and pattern recognition (or machine learning) module. In our study, different representation features borrowed from the natural language processing (NLP) are investigated and a set of machine learning techniques commonly used for similar task are also investigated for our proposed task.

3.1 Features Reduction

By reviewing good number of related studies, as described in the literature review earlier in this report, we have seen that most of the reviewed studies used features commonly and successfully used in the NLP. The most common features are the n-gram language models at words level. By this, each review is represented by a vector of features. For uni-gram, a vector of unique words appears in the whole training data and for each review, the frequencies of words appear in the review are used as features.

Since the number of unique words is high compared with the number of words of a single review, most of the frequencies in the features vectors are zeros except the words consisting the review. The Bi-gram is similar except it looks at the frequency of sequence of two words, i.e. the probability of two words coming with each other. In this case, the feature vector size is squared of the number of unique words appeared in the training data. This huge feature dimension required a lot of training data.

TABLE 3.1: N-Gram Structure of User's Reviews.

User's review	**Uni-gram**	**Bi-gram**
Scam app just for stealing data	"Scam"	"Scam app"
	"app" "just" "for" "stealing" "data"	"app just" "just for" "for stealing" "stealing data"

As shown on the Table.3.1, we show some examples of uni-gram and bi-gram after applying it on the original user's review that selected from data set.

3.1.1 TF-IDF Features

The TF-IDF (term frequency-inverse document frequency) features are statistic based features that reflect the importance of a word across a set of documents. This measure is composed of two individual measures: the term frequency and the inverse of the document frequency.

Words in the document with a high tf-idf score occur frequently in the document and provide the most information about that specific document.

In most cases, the higher the occurrence of a word to appear in a document the bigger the TF coefficient. That means both are directly proportional. Importance coefficients is inversely proportional of these two, as it goes higher as the word occurrence go lower. With that being said, to compute TF*IDF, one must know the value of term occurrences.

TF*IDF is an important technique in showing the comparison of the term's frequency (TF) and Inverse document frequency (IDF). Each word or term has its own respective TF and IDF score. Multiplying these two scores of a term, we get the product called TF*IDF weight of the terms.

It is safe to assume that the higher the TF*IDF score (weight), the lower the term and vice versa.

The TF*IDF shows and weighs down the number of occurrence of that specific word in a document and gives essential points to that word based on the number of times it appears in that document. Aside from that, it also provides an information on how important a word throughout the whole article, which is referred to as corpus.

For a term t in a document d, the weight Wt,d of term t in document d is given by:
Wt,d = TFt,d log (N/DFt) Where:

- TFt,d is the number of occurrences of t in document d.
- DFt is the number of documents containing the term t.
- N is the total number of documents in the corpus.

3.1.2 Noise Removal

Processing noisy data may cause serious problems on machine learning applications during the training process, as well as the testing process. so when we removed the noisy words, we reduce memory consumption, prevent over-fitting and improve speed.

In our experiments, the extra spaces, symbols and numbers are removed from the user's review before applying the further processing.

TABLE 3.2: Example of the user's review before and after removing the noise

Original review	Review After removing noise
"6/8/17 is the last update today I got a notification to update problem today is 6/14/17!!!!!!(NOT DEPENDABLE) TRACKING FOR SOME SECTIONS When I am in a taxi stupid app tracks movement as riding a bicycle. DEVELOPMENT HUMONGOUS ERROR OR TESTERS NEGLECT!"	is the last update today I got a notification to update problem today is NOT DEPENDABLE TRACKING FOR SOME SECTIONS When I am in a taxi stupid app tracks movement as riding a bicycle. DEVELOPMENT HUMONGOUS ERROR OR TESTERS NEGLECT

As shown on the Table.3.2, we show some examples of review after remove the noisy words from the original user's review that selected from dataset.

3.1.3 Stop words removal

Stop words are the words ,which are filtered out before or after processing of natural language data (text). Though "stop words" usually refers to the most common words in a language, and that show up a lot in every document (e.g. prepositions and pronouns).

TABLE 3.3: example of the user's review before and after removing stop words

Original review	Review After removing stop words
"** Galaxy S5 ** The product is built around the information in the app, but I've just installed it and so far couldn't activate my account in the app and now I can'tsetmy target weight. The graphics are allmessed up and restarting app and phone does nothing. Doesn't seemlike they tested much beforerelease. Will update review once the apps updated."	"** Galaxy S5 ** The product built information app, I've installed far activate account app I set target weight. The graphics messed restarting app nothing. Doesn't like tested release. Will update review apps updated."

As shown on the Table.3.3, we show some examples of review after remove the stop words from the original user's review that selected from dataset.

3.2 Machine Learning Techniques

After extracting representative feature vectors from the text reviews of length N for Uni-grams and N2 for Bi-grams, by the feature extraction module, a machine learning technique is used to build a model for each class using the extracted feature vectors from the training dataset. By reviewing the related studies [32] [9] [25] in the field, the most common machine learning techniques used are:

- Naïve Bayes
- Multinomial Naïve Bayes
- Random Forest

- Support vectors machines (SVM).

All of these techniques implemented in the well-known Weka toolkit [39]. Therefore, it is used in all of our experiments represented in the following chapters.

In order to build a model for each class with specific machine learning technique, we need to train the model with a training data for each class. To evaluate the accuracy of the trained system in recognizing the true class of a given review, a subset of the data set, which doesn't appear in the training data need to be used for the testing purpose. For this, the data set was divided into two non-overlapped subsets; one for training the system and one for testing it. The training data set is around two-thirds the whole data and one third is left for the testing. More details of the dataset and training/testing divisions are found in the data collection and analysis section.

The proposed methodology needs to be validated by reporting the performance of the automatic systems using performance measures that used in this field such as accuracy, recall, precision, and F-measure. We also aim to evaluate and compare various machine learning techniques in order to choose the best model for this specific task.

So the Figure.3.1 shows the proposed steps that we should follow them passed the review into different sub-systems in order to predict its type, the front end system classify the reviews in to Bugs, New feature or Sentimental, after that we pass the user's review that our system classified as Bugs in order to get more

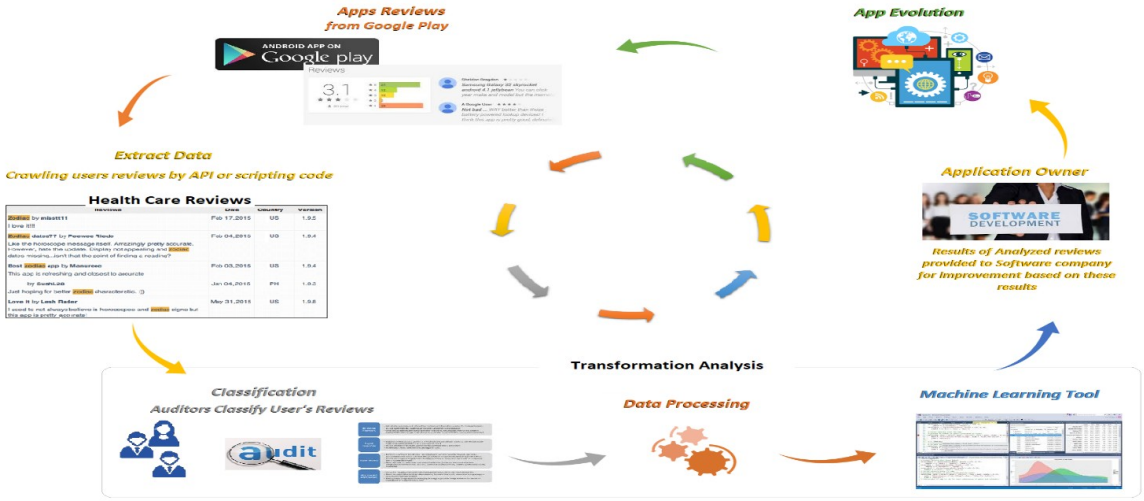

FIGURE 3.1: Architecture of the reviews classifications

specific classification, so that back end system classify the Bugs into four categories, which are General Bug, Security problem, usability problem or performance problem, also we build the sentimental analysis system , which get specific classification of the sentimental, theses specific classifications are positive, negative or neutral, the Figure. 3.2 show these detail

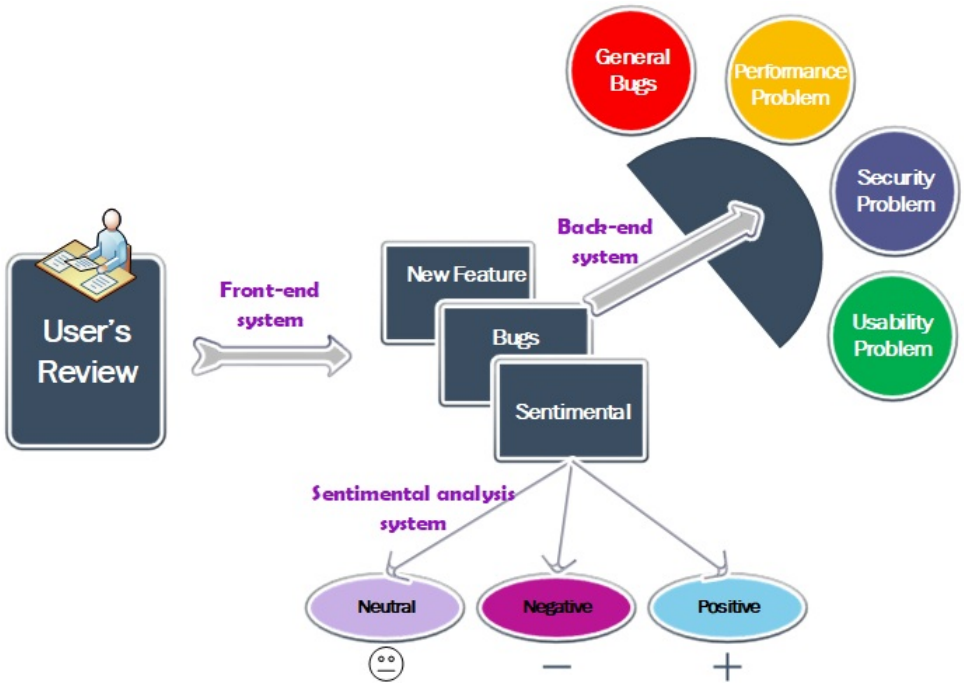

FIGURE 3.2: Architecture of the reviews classifications

Chapter 4

Data Collection and Analysis

4.1 Data Collection

To achieve our objectives and implement our described system in chapter 3 earlier, a suitable and sufficient data set is needed. To our knowledge, there is no available dataset that can fit our target in this project. Therefore, we decided to collect our own dataset and do the required analysis and annotation according to our needs. The most challenging task of the data collection is the labeling of each review by an expert (or more) according to the classes we defined in this project. To make this easier, we developed a web site by, which volunteer experts can read a randomly selected set of reviews and give some details about each review such as the impression of the user (sentimental), assign it to one of our five target classes if possible. Since the five classes labeling is the most important for our project, we also asked the expert to express his/her confidence level of his/her decision (high, medium or low confidence). This will help us train our systems on data labeled with high confidence and find the correlation between the system predictions and the expert labels. We are also interested in

comparing the performance of systems trained on data labeled with high confidence and medium and low confidence. The data collection process is described in the following stages:

1. Retrieve a sufficient number of user reviews on mobile applications in the field of healthcare. We have retrieved around 90,000 reviews written in English for 10 healthcare applications from Google Play Store. The Figure. 4.1 below shows the selected 10 applications with their average rating. The figure above reflects the distribution of the average of rating of

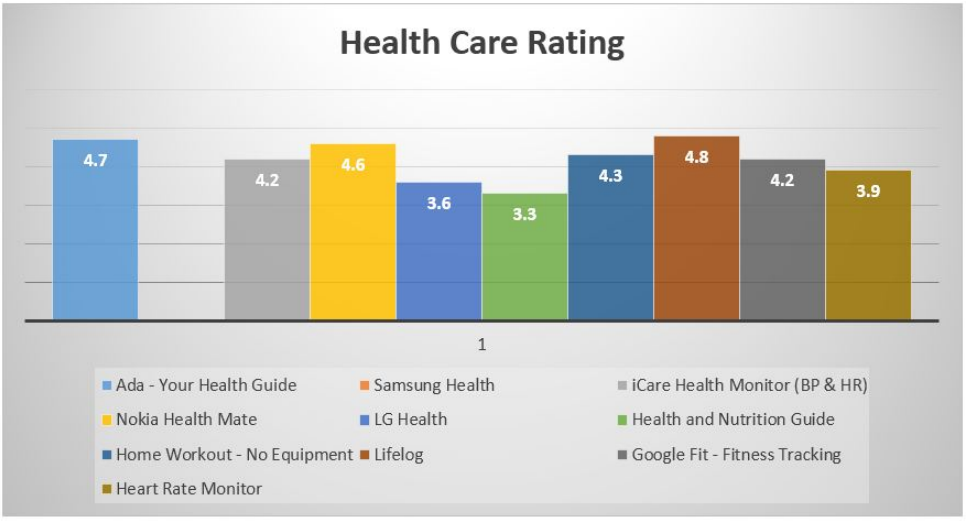

FIGURE 4.1: Healthcare applications with their overall ratings

healthcare applications by did by users, we note from the figure that the highest average rate is for "Ada-your health Guide-Google play", and the lowest rating is for Health and nutrition guide. We developed a python code for this purpose, and we used Chrome selenium driver in order to collect this number of reviews. This process was very exhausting since we were monitoring the system while data gathering, which took relatively long time. We created a database for this huge data including review texts, app names, submission dates, usernames, and rating stars.

2. The collected reviews need to be annotated according to what we have defined in the methodology. The annotation process is a manual process and needs to be done by expert people with a good experience in the software development and maintenance (Software engineering). It is difficult (or impossible) to ask one expert to annotate this huge number of reviews. Therefore, 10 experts with different experience participate in this process. To make it for the experts easy, we built a website using PHP, Apache, and MySQL for this purpose. We published this website through the internet and asked volunteers who have good experience in the field of software development and engineering, to help in the annotation process. This can be done remotely over the internet by first filling in a simple form describing his/her experience in software, number of years working in this field, age, contact information, work and resident location, gender and the specific field of experience. By filing this form, the participant will get a username and password by, which he/she can get access to the online system (website) where he/she can start the annotation process. Once login, the participant gets a randomly selected number of reviews (default number is 100 but can be more if the expert is willing to do more). For each review, the expert needs to express the reviewer's impression about that application as positive, negative or neutral (for sentimental analysis). Also, the expert is asked to classify the presented review as one of our five target classes (Bug, New Feature, Performance, Security, Usability or Sentimental), if this review doesn't belong to any of the specified classes. In addition to this, the expert is asked to express his/her confidence level for each of this classification, as a high, medium or low confidence. This will help us to identify the reviews labeled with high confidence and that with less confidence, and study the effect of this on our proposed systems.

In some cases, we had meetings with some of the expert participants prior to the annotation to illustrate them more about the idea of this project and make sure they understand what they need to do carefully. Even with a good and related experience, the annotation process is a subjective decision, having more labeling decision for each review from multiple experts gives more confidence to the final decision. For example, if two experts label the same review with the same class label, this means that most probably the label is correct. If three experts label the same review, and two agree on the same label, but one had a different label, most probably the two experts label is the correct one. It is not possible to have multiple (two or more) for each review, because it needs either each expert to label many reviews or to have many experts working on the same reviews. To overcome this limitation, we select the sample reviews for each expert randomly. By this, some of the reviews will have labels by two experts. Then, we can easily select a subset of the annotated data consists of reviews with two or more annotators agree on the same label. Furthermore, we can have a subset consists of reviews have the same label by two annotators with high confidence decisions.

Some of the presented experiments in the next chapter were conducted on different data subsets as explained above to study the impact of labeling confidence on the system performance.

In our experiments we exclude all reviews that have many classifications on the same time. For example, we excluded the review that classified as security and usability, or Bug and performance. On the remainder section of the data collection, we present some examples of the collected reviews with their annotation by the experts and the sentimental analysis of them.

TABLE 4.1: Sample of the data classified as bugs including the sentimental of the review

Review	Review Classification	Review Impression	Rating
Suddenly stopped working even after being to the gym for 2 hrs saying I've only done 9 steps	Bug	Negative	3

The Table.4.1 shows a sample of the data classified as bugs, also the table includes the sentimental of the review in general.

TABLE 4.2: Sample of the data classified as performance including the sentimental of the review

Review	Review Classification	Review Impression	Rating
I generally love this app. But, after the latest update, it has been running extremely slowly on my Galaxy S7. It also freezes for seconds to nearly minutes at a time before finally syncing my steps, food, exercise, etc.	Performance	Negative	3

TABLE 4.3: Sample of the data classified as security including the sentimental of the review

Review	Review Classification	Review Impression	Rating
Scam app just for stealing data	Security	Neutral	1

The above Table.4.3 shows sample of the data classified as a security problem, also the table include the sentimental of the review in general, it's clear that some user's review rated as five but they asked us to solve some problems in the software.

TABLE 4.4: Sample of the data classified as Usability including the sentimental of the review

Review	Review Classification	Review Impression	Rating
Doesn't show steps recorded on devices between iOS and android. Pretty annoying as both sync to the same service.	Usability	Negative	2

The above Table.4.4 shows a sample of the data classified as usability problem, also the table includes the sentimental of the review in general.

TABLE 4.5: Sample of the data classified as new feature including the sentimental of the review

Review	Review Classification	Review Impression	Rating
I really liked it. Currently tracking all my activities. The one thing they need to add is "share my running feeds" ,which i can display to everyone, may be in blog, Google+. May be some kind of widget... That would be a killer...	New Feature	Positive	4

The Table.4.5 shows sample of the data classified as a New feature, also the table include the sentimental of the review in general.

TABLE 4.6: Sample of the data classified as multiple classifications including the sentimental of the review

Review	Review Classification	Review Impression	Rating
This ap does not count steps correctly when challenging others. Sometimes its over and sometimes under. Makes challenging friends useless	Bug,Usability	Neutral	2

The Table.4.6 shows sample of the data classified that the review contains multiple classifications, also the table include the sentimental of the review in general.

4.2 Data Analysis

In this section, we present some analysis on the collected and labeled data, in terms of number of labeled reviews for each of our target classes, number of reviews labeled with high confidence, medium and low confidence. Also, the number of reviews labeled by only one expert and that labeled by two or more experts.

4.2.1 Sentimental Analysis

In total, 7613 reviews were labeled by at least one expert according to the user impression as positive, neutral or negative.

TABLE 4.7: Customers rating for medical applications

Count of reviews	Rating
1817	1
865	2
1097	3
1166	4
2668	5

The Table.4.7 shows total number of reviews for each user's rating, so our collected data distributed over all rating numbers, by random study of sample data we observed that the user's complaints are not limited with rate one and two, but it always exists in rate four and five, this mean that the user is love the applications but it has some problems or need new feature.

TABLE 4.8: Statistical analysis of the classified review

Positive	Negative	Neutral
3630	2766	1217

The Table.4.8 shows the number of classified reviews belongs to the sentimental categories. Also we did some statistical analysis of the annotation process of the collected data, as shown in the below table, it shows the total number of annotated reviews for each of the five target classes. For each class, the number of annotated reviews with high, medium and low confidence is also shown in the table. Moreover, for each class, the number annotations, which are done by one expert and the number of annotations, which are done by two experts.

TABLE 4.9: Statistical analysis of the classified review

Requirement Classification	Review Classification count	High confidence	Medium confidence	Low confidence	Unique reviews	No. reviews auditing by 2 auditors
Bug	2758	2205	514	39	1513	867
Usability	839	638	186	15	666	142
New Feature	1170	889	257	24	748	308
Performance	555	383	164	8	465	72
Security	96	62	27	7	71	17

Though our observation of the data in the Table.4.9, its shows that Bug category represent most of the data then new feature, usability, performance and security respectively. Also the data shows that the percentage of the high confidence was greater than the medium and low, this lead that the in general the user express well on his/her need of the software.

TABLE 4.10: Classification of the user impression with relative to real rating did by the user himself

No.reviews	**review impression classified by experts**	**User's rating**
1585	Negative	1
653	Negative	2
566	Negative	3
232	Negative	4
148	Negative	5
385	Neutral	1
240	Neutral	2
323	Neutral	3
249	Neutral	4
187	Neutral	5
100	Positive	1
79	Positive	2
362	Positive	3
829	Positive	4
2454	Positive	5

The Table.4.10 shows our classification of the user impression with relative to real rating did by the user himself, so we group the data set to the user's impression and rating, so we classify the count of the reviews for each impression relative to the rating and it's clear that even so the rating of the users was four or five their comments were negative, also some users rated the application as one and their review was positive.

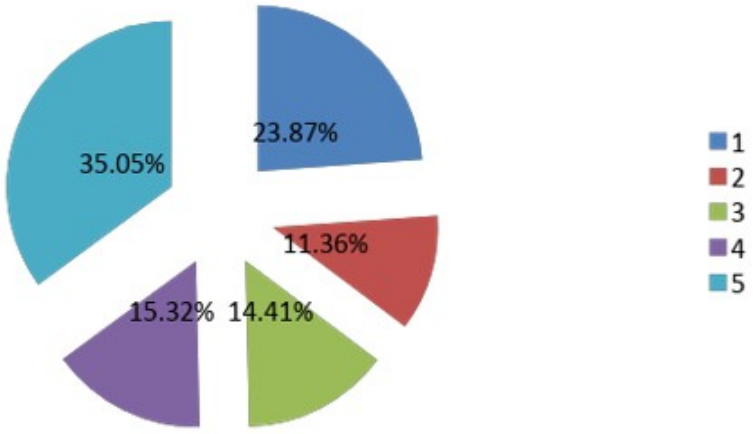

FIGURE 4.2: Percentage distribution of the customer's reviews rating

As shown in the Figure. 4.2 we distributed the rating as a percentage, we note that the highest percentage is for the users who rate 5, the second is for who rate 1, the third is for who rate 4, the fourth is for who rate 3 and the least percentage is for who rate 2.

As shown in Figure.4.3 , the majority of users classified the healthcare applications positively by 47.68%, where the second rate classified them negatively by 36.33% and only 15.99% classified them neutrally.

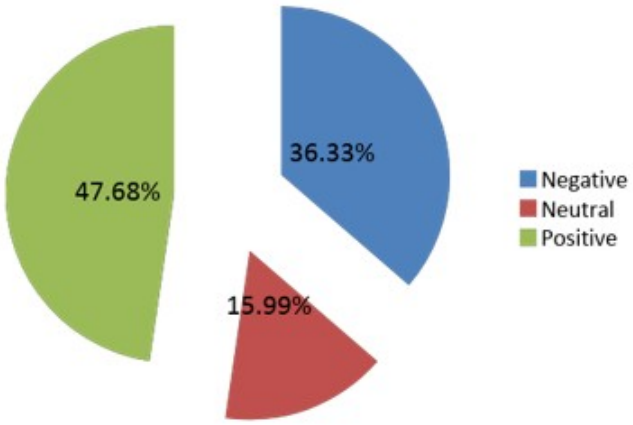

FIGURE 4.3: Distribution of Reviews by classification

Chapter 5

Experiments and Results

5.1 Experimental Setup

After labeling a sufficient part of the collected data by expert volunteers and doing some analysis and pre-processing on the annotated data, as presented in the earlier chapters, this chapter describes the experiments we did on the annotated data and present the results we achieved.

Different classifications techniques are implemented and investigated in order to see ,which one is the best for extraction the Non-functional requirements (NFR), based on the literature review we did in the field of our study, we decide to use the following classification techniques in our systems.

1. Naïve Bayes
2. Naïve Bayes Multi nominal
3. Random Forest
4. Support Vectors Machines

These classifiers are used to solve multi-class classification tasks, also these are the most popular and accurate multi-class classification methods in the given research domain. So we introduce a description about the used algorithms:

1. Naïve Bayes:

   The Naive Bayes [10] is a supervised classification algorithm based on Bayes' Theorem, assuming that the class has unrelated features, hence the word naive. The Naive Bayes classifier calculates the probabilities for every factor; where the result of highest probability has been chosen
   The theorem is described in Figure.5.1 (A), the prior probability of A ,which means it does not consider any information about B. (A | B), the conditional probability of A given that B is true .

   A Naive Bayes model assumes that each of the features it uses are

$$P(A|B) = \frac{P(A)\,P(B|A)}{P(B)},$$

FIGURE 5.1: Bayes' theorem

   conditionally independent of one another given some class [37], so a document is considered to be an ordered sequence of words obtained from vocabulary.

   Naïve Bayes classifiers have been successfully applied in many domains such as

   - Classifying the email is spam or not
   - Classify a news article into different categories such as: technology, politics and sports.

- To check if a piece of text is expressing positive or negative emotions.
- Being used for face recognition software.

2. Multi nominal Naïve Bayes:

   The Multinomial NB implements the naive Bayes algorithm for multinomial distributed data, and it is one of the two classic naive Bayes variants used in the text classification where the data are typically represented as word vector counts [38], Where the Multinomial Naïve Bayes Classification is the number of occurrences of "t" in training documents from class c, it also includes multiple occurrences. [37]

   Multinomial Naive Bayes assumes multinomial distribution for all the pairs, which seem to be a reasonable assumption in some cases, i.e. for word counts in documents.

   Random Forest:

   Random Forest is a supervised learning algorithm, it's considered a powerful algorithm proposed [10], which combines several randomized decision trees and aggregates their predictions by averaging, it creates a set of decision trees from randomly selected subset of training set, most of the time trained with the 'bagging' method, this adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features.

   The algorithm is considered great for early training in the model development process, to see how it performs. It is hard to build a "bad" Random Forest; because of its simplicity. This algorithm is also a great choice, if

you need to develop a model in a short period of time.

Random Forests are considered very hard to beat when it comes to performance and time[6]. There are other models that can perform better, such as the neural network, but at the cost of time in development. In addition to that, they can handle a lot of different feature types, like binary, categorical and numerical.

Support Vectors Machines:

Support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. [16] it comprises of a set of related supervised learning methods that is used for classification and regression.. In simple words, given a set of training examples, each marked as belonging to one of two categories, the SVM training algorithm builds a model that predicts whether a new example falls into one category or the other. [16]

SVM is widely used in a wide range of applications that includes but not limited to content-based image retrieval, biometrics, object detection and recognition, text recognition, speech recognition, speaker identification, and benchmarking time-series prediction tests. [20] [8]

In total, we have about 7600 raw reviews. We manually labeled 3600 unique reviews that were classified as non-functional requirements or new features and sentimentally like a negative, positive or neutral. Of these, around 1700 reviews were annotated by two experts, and the rest by only one expert. Some of the presented experiments are conducted on the high confidence labeled data and

some on the whole data (i.e. three levels of confidence).
The purpose of these experiments is to study the impact of expert confidence, when doing data annotation, on the classification system performance. Some other experiments are conducted to study the impact of doing the annotation by only one expert and multiple experts. Intuitively, when two or more experts agree on the same label for a specific review, it implies that this label is most probably correct. On the other hand, when only expert does the labelling or two experts or more but with contradiction, the probability that the annotations are correct is lower.

In addition, for these experiments, we excluded all reviews that have contradicting labels on the same time. for instance, we excluded the reviews that classified as security and usability, or Bug and performance by different experts. To exploit all information and properties of the collected data, we took the large sub set of data the matches with the criteria below and we consider it as base line criteria, after that we changed some parameter of the criteria to compare the new result with base line criteria result.

Based on the literature review [25] [18] [7] we see that to apply Bi-gram, remove stop words and remove noisy words and TF-IDF, so that we made these parameters are constant in all experiments. Also we exclude the contradiction reviews and reviews that has many classifications

The following pre-processing steps were applied on the raw reviews before executing the experiments.

1. WEKA Toolkit [39] was used in all of our experiments presented in this thesis.

2. The raw reviews were converted to arff format to be compatible with Weka toolkit.

3. Each review was converted to the word of the vectors using an unsupervised filter (StringToword vectors) with BIgram features and other default attributes on this filter type , Figure.6 in the appendix shows the applied settings in the experiments .

4. Apply the BI-gram

5. Remove the defined stop words and noisy words

6. Apply TF-IDF

7. The above mentioned four ML classifiers (Naïve Bayes, Multi nominal naïve Bayes, randomly forest and SMM) were trained on the training subset data (66% of the used data) and evaluated on the testing subset data (34% of the used data), also we apply the cross validation on the data set.

   We aimed from this step to see the effective of the trained/tested data to the classification accuracy, so that we take a sample of 66% for our data to train on the data set and we use 34% of the sample to test our data set, of course the data selected randomly, another familiar and sophisticated way that we use it is cross validation techniques. we use k-fold cross-validation method K-fold cross-validation works by using part of the data to train the model, and the rest of the data set to test the accuracy of the trained model.

   We use 5 and 10 folds. The goal is to have a good balance between the size and representation of data in your train and test sets The cross validation chooses the subset randomly by shuffled sampling type.

8. We run the experiments on the different subsets to see the effect of the data on the classifier accuracy.

9. We run the experiments after applying the resample techniques to minimize the biasing on the data and we record the result and compare the result with experiments result before applying the resample techniques The Figure. 5.2 shows these steps.

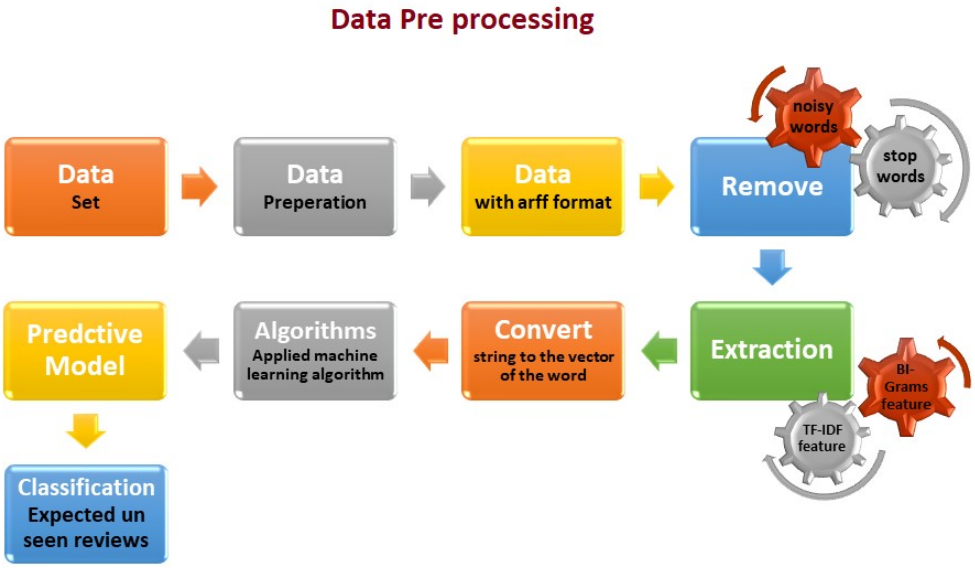

FIGURE 5.2: Data set preprocessing steps

5.2 Baseline system

In order to study the effectiveness of some techniques and data quality and quantity, we developed a baseline system, which will be used as a reference system for all proposed systems and experiments presented in this thesis.

5.2.1 Baseline configurations

In all of our experiments, well-known Weka toolkit has been used for implementing our proposed systems. All raw reviews done by one or more experts (with high, medium and low labeling confidence) have been divided into two subsets; 66% train and 34% test. All reviews with contradiction labels and that have more than one label were excluded from the training and testing datasets.

The training data was used for training the baseline system, where testing data was used for evaluation. The words of each review were converted to feature vectors using bi-gram TF-IDF without removing stop/noisy words.

The word-level TF-IDF bigrams vectors extracted from the training data were used to train different classifiers (mentioned in section 5.1); Naïve Bayes, Multi nominal naïve Bayes, randomly forest and SMM. The feature vectors extracted from the testing data were used to evaluate the resulting classifiers. The system performance was represented by accuracy, precision, recall and F-measure percentage.

5.2.2 Baseline Performance

As explained in the methodology chapter in this thesis, our overall system consists of two pipelined modules (or sub-systems). The front-end system takes the input review and classifies it as one of four major classes; bug, new feature request, sentimental and others. The back-end module takes all of the reviews that are classified as bug from the front end system , and classifies them further as general bug, security defect, and performance problem and usability difficulty.

5.2.2.1 Front-end module results

Table.5.1 shows the performance of the front-end module of our baseline system, for the four chosen classifiers.

TABLE 5.1: Performance results of the front-end system

Classifiers	Accuracy				Avg. Precision	Avg. Recall	Avg. F-Measure
	Correctly Classified Instances		Incorrectly Classified Instances				
Naïve Bayes.	931	69.48%	409	30.52%	0.69	0.69	0.69
Multinomial Naïve Bayes	977	72.91%	363	27.09 %	0.74	0.72	0.73
Random Forest	968	72.24%	372	27.76%	0.73	0.72	0.69
SVM	959	71.57%	381	28.43%	0.71	0.72	0.71

As shown from the results, the performance of the four classifiers are relatively close with the Multinomial Naïve Bayes has the best performance, 73% accuracy and F-score. The results presented in Table.5.1 are the overall performance for the three classes. It is worth to mention here that the Multinomial Naïve Bayes machine learning technique has been used in all of the experiments described in the following sections and chapters.

In order to study, how well this system in predicting each class, we compute the performance of each one, as shown in the results presented in Table.5.2 . In general, the system ability to recognize sentimental reviews and reviews talking about bugs or problems in the applications is significantly better than recognizing reviews that requesting new features.

TABLE 5.2: Per-class performance of the frond-end system

Class / Classifier	Bug			New Feature			Sentimental		
	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
Naïve Bayes	0.67	0.68	0.67	0.49	0.52	0.50	0.78	0.76	0.77
Naïve Bayes Multinomial	0.68	0.78	0.72	0.52	0.52	0.52	0.85	0.75	0.90
Random Forest	0.66	0.77	0.71	0.81	0.11	0.19	0.76	0.87	0.81
SVM	0.67	0.67	0.67	0.58	0.41	0.47	0.77	0.84	0.81

TABLE 5.3: Front-end system Confusion Matrix with Random Forest

True class / Recognized	BUG	New Feature	Sentimental
BUG	374	48	59
New Feature	61	100	32
Sentimental	117	46	503

These results show the system performance in recognizing each class, but do not show the system confusion among these classes. For example, the percentage of reviews requesting new features that are falsely classified as bug and that classified as sentimental. By computing these figures, we can see the classes that have the most confusion and that have the less. Confusion matrix is very powerful way of presenting all of these details. For each class, It shows the number of reviews correctly classified and that falsely classified as one of the other two classes. Table 5.3 and table 5.4 show the confusion matrix of the best two classifiers, Random Forest and multi nominal Naïve Bayes, respectively.

TABLE 5.4: Front-end system Confusion Matrix with Random Forest

True class / Recognized	BUG	New Feature	Sentimental
BUG	371	2	108
New Feature	101	21	71
Sentimental	87	3	576

It is clear from the figures in the two confusion matrices shown in Table 5.3 and table 5.4 that the confusion is higher between the new feature class with both bug and sentimental classes. Although the overall performance of the multi nominal Naïve Bayes system is slightly higher than the random forest, its confusion between the new feature and the two other classes is significantly high compared with the Random Forest based system.

5.2.2.2 Back-end module results

The backend system takes the reviews that are classified as bug by the front-end system and classifies them further as one of predefined four classes, namely, general bug, security defect, performance problem and usability difficulty. In our collected dataset, experts classified 1600 reviews as bugs. Out of these, 1162 are general bugs, 45 security bugs, 126 performance bugs and 269 usability bugs.

The back-end system classified 376 reviews correctly out of 545 testing reviews, i.e. 69% accuracy percentage and 68% F-score, as shown in Table.5.5

TABLE 5.5: Back-end system performance

Classifiers	**Accuracy**				**Precision**	**Recall**	**F-Measure**
Multinomial Naïve Bayes	**Correctly Classified Instances**		**Incorrectly Classified Instances**				
	376	68.99%	169	31.01%	0.68	0.69	0.68

By looking at the system performance for recognizing each class, as shown in Table.5.6 below, we find that the system is biased to the general bugs class with accuracy of 84% compared with 36%, 12% and 8% for the usability, performance and security, respectively. Our justification for this result is that because the

TABLE 5.6: Details Accuracy of the classification techniques for base Line criteria-stage2

	General Bug			Security			Performance			Usability		
Naïve Bayes Multinomial	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
	0.84	0.89	0.86	0.08	0.27	0.12	0.12	0.09	0.11	0.36	0.14	0.21

TABLE 5.7: Back-end system confusion matrix

Classified as \ True class	General	Security	Performance	Usability
General	355	9	20	15
Security	3	4	3	5
Performance	22	12	4	3
Usability	44	26	7	13

available training data for the security, performance and usability are very small compared with the General bug data, as mentioned at the beginning of this subsection. As expected, the system confusion is high between the general bug class and the other three classes, as shown in the confusion matrix in Table.5.7 below. There is also noticeable confusion between usability and security.

5.2.2.3 Sentimental Analysis Results

As described in the data analysis section, 2258 of the collected reviews includes only user personal opinion expression about that particular application as a positive expression, neutral or negative. Eighty-one percent of these reviews express positive opinions, 12% neutral opinions, where the rest express negative opinions. This type of reviews do not include any functional or non-functional software requirements. As explained earlier in this chapter, our front-end system identifies sentimental reviews. In order to recognize how many of these reviews praise of the particular application, how many of these are neutral and how many of these reviewers are not happy with the application.

TABLE 5.8: Per Class performance of the sentimental analysis system

	Negative			Positive			Neutral		
Naïve Bayes Multinomial	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
	0.63	0.59	0.61	0.93	0.93	0.93	0.324	0.35	0.34

TABLE 5.9: Confusion matrix of the sentimental analysis system

True class / Classified as	Negative	Positive	Neutral
Negative	56	16	23
Positive	18	565	25
Neutral	15	27	23

Using the same configurations of the previous described systems, a sentimental recognition system was trained and evaluated on the reviews labeled as sentimental. Three –class Multi nominal naïve Bayesian system, trained on 66% of the available data, classifies each sentimental review as positive, neutral and negative. The remaining 34% reviews were used for system evaluation. The overall performance result is shown in the table 5.10, where the per-class performances were shown in Table. 5.8. The overall accuracy is about 84% with 84% F-score. Since the training data for each class is not balanced, the system accuracy for recognizing each class is higher for the class with more data. The system recognizes positive reviews with 93% accuracy, where it recognizes negative and neutral reviews with accuracy 63% and 32%, respectively. In the subsequent sections, we treat this imbalance in the dataset with different methods.

TABLE 5.10: Overall performance of the sentimental analysis system

Classifiers	Accuracy		Precision	Recall	F-Measure
Multinomial Naïve Bayes	Correctly Classified Instances	Incorrectly Classified Instances			
	644 83.85%	124 16.15%	0.841	0.84	0.84

As shown on the confusion matrix in table.5.9 , the system falsely recognizes neutral reviews as positive and negative equally.

5.3 Study the effect of data label confidence on the system performance

In this section, we study the effect of confidence level of the experts when manually labeled the reviews. In all of the previously presented experiments, all reviews with the three confidence levels were used. This implies that some of the used reviews have wrong labels especially those with low and medium confidence level. This surely degrades the system performance and increases the system confusion between the target classes. Now, in order to study in detail how bad this affects the system performance, we kept the same configurations of the baseline system except the dataset. Only the reviews labeled with high confidence were used in training and testing the baseline system. This results in 3699 reviews; 1379 bugs, 451 new features and 1869 sentimental reviews.

TABLE 5.11: Per-class performance of the front-end system when using high confidence reviews

	Bug			New Feature			Sentimental		
	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
Baseline (using all dataset)	0.68	0.78	0.72	0.52	0.52	0.52	0.85	0.75	0.90
Using high confidence reviews	0.80	0.85	0.82	0.71	0.58	0.64	0.87	0.87	0.87

5.3.1 Front-end system results

Table.5.12 below shows the front-end system performance. The overall system accuracy and F-score are improved from 73% (front-end baseline system) to 83% when using reviews labeled with high confidence. This 13.7% improvement is expected, since the number of reviews with wrong labels are minimized in the used dataset.

TABLE 5.12: Front-end overall accuracy when using high confidence reviews

Classifiers	**Accuracy**		**Precision**	**Recall**	**F-Measure**
Baseline (all dataset)	**No of Correctly Classified Instances with accuracy**	**Incorrectly Classified Instances**			
	977 72.910%	363 27.09%	0.74	0.72	0.73
only high confidence data	1040 82.67%	218 17.33%	0.83	0.83	0.83

It is also clear that the recognition rate of the three classes are improved. The sentimental class remains at the top with the highest recognition rate, 87%.

5.3.2 Back-end system results

As mentioned earlier, the backend system classifies the bug reviews into four sub-classes. By applying the high confidence filtration on the available reviews labeled as bugs, we end with 1459 reviews that labeled as bugs with high confidence. This data only will be used to train and test the backend system. Of this data, 1102 reviews talk about general bugs, 230 talk about usability bug, 95 performance bugs, where only 32 reviews talk about bugs in the security.

TABLE 5.13: Backend overall performance when using high confidence data

Classifiers	Accuracy		Precision	Recall	F-Measure
Baseline (all dataset)	**No of Correctly Classified Instances with accuracy**	**Incorrectly Classified Instances with accuracy**			
	376 68.99%	169 31.01%	0.68	0.69	0.68
Using high confidence data	385 77.62%	111 22.38%	0.79	0.78	0.72

As shown in Table.5.13 above, the overall accuracy is about 78 %, whereas, the accuracy of the backend system in the baseline is 69 %. This means that using the reviews labeled with confidence improves the backend system by 13%. This improvement is very similar to the improvement achieved by the front-end system described earlier.

In general, the recognition rate of all classes is improved when we used high confidence data. The general bug category has the best rate with about 0.78

TABLE 5.14: Per-Class performance of the backend system when using high confidence data

	General Bug			Security			Performance			Usability		
Baseline(all dataset)	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
	0.84	0.89	0.86	0.08	0.27	0.12	0.12	0.09	0.11	0.36	0.14	0.21
Using high confidence data	0.78	0.99	0.87	1.00	0.08	0.14	1.00	0.11	0.19	0.74	0.24	0.37

for the precision, 0.99 for the recall and 0.87 f-measure. This is because of the amount of data for each class is not equal. This will be investigated experimentally in the following sections.

5.3.3 Sentimental analysis system results

By applying the same high confidence criteria on the sentimental reviews, we end up with 2035 reviews; 1763 positives, 197 negatives, and 75 neutrals. Similar to the frond-end and backend systems, the overall and per-class results of sentimental analysis system have improved significantly as shown in tables 5.15 and 5.16. The overall system accuracy is improved by around 7% (90% accuracy compared with 84% for the baseline system). By looking at the per-class results in table 5.20 below, we notice that the recognition rate for the positive class is improved slightly, whereas, the recognition rate for the other two classes (negative and neutral) is decreased. This, maybe, because of the reduction for data of these two classes after applying the high confidence filtration.

TABLE 5.15: Overall performance of the sentimental analysis system

Classifiers	Accuracy		Precision	Recall	F-Measure
Baseline (all dataset)	**Correctly Classified Instances with accuracy**	**Incorrectly Classified Instances**			
	644 83.85%	124 16.15%	0.841	0.84	0.84
Using high confidence data	624 90.17%	68 9.83%	0.87	0.90	0.88

TABLE 5.16: Per Class performance of the sentimental analysis system

	Negative			Positive			Neutral		
Baseline(all dataset)	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
	0.63	0.59	0.61	0.93	0.93	0.93	0.324	0.35	0.34
Using high confidence data	0.85	0.40	0.55	0.91	1.00	0.95	0.01	0.01	0.01

5.4 Study the effect of number of experts on the system performance

In order to study the effect of having two experts or more and all have agreed on the data labeling, we apply new filtration criteria on the dataset to select only the reviews with two or more experts and all agree on the review classification labels. By having multiple expert's agreement on the same review labels, we guarantee the quality of the data labels. Hence, the automatic system is trained and evaluated on robust data. Out of this process, we end up with 1474 reviews

(875 bugs, 343 new features and 283 sentimental). This data was used to train and evaluate front-end sub-system of our baseline system.

The following tables show the overall accuracy and per-class results of our three baseline sub-systems (front-end, back-end and sentimental). It is worth to remind the reader that all of the experiment configurations are kept the same for the baseline system, such as classifier type, data division (66% training and 34% testing), and other parameters.

TABLE 5.17: Overall accuracy front-end system using data labeled with two or more experts

Front-end Baseline (all dataset)	**Accuracy**		**Precision**	**Recall**	**F-Measure**
	No of Correctly Classified Instances with accuracy	**Incorrectly Classified Instances**			
	977 72.910%	363 27.09%	0.74	0.72	0.73
only high confidence data	1040 82.67%	218 17.33%	0.83	0.83	0.83
Two or more experts	382 76.25%	119 23.75%	0.78	0.76	0.75

As shown in the Table.5.17, the overall accuracy is about 76% whereas, the accuracy of the baseline is 73 %. As we expect, that multiple experts have positive impact on the system performance.

TABLE 5.18: Per-Class performance of front-end system using data labeled with two or more experts

Front-end	Bug			New Featur			Sentimental		
Baseline(all dataset)	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
	0.68	0.78	0.72	0.52	0.52	0.52	0.85	0.75	0.90
Using high confidence reviews	0.80	0.85	0.82	0.71	0.58	0.64	0.87	0.87	0.87
Two or more experts	0.76	0.95	0.84	0.66	0.49	0.56	0.94	0.49	0.64

As shown in Table.5.18, the sentimental class has the best precision, where the accuracy was about 0.94 for the precision. Then, it comes the bug as the next better class with precision of about 0.76, where, the precision the new feature class is 0.66.

For the back-end system, about 770 reviews were resulted from the filtration process with two or more experts agreed on the data labels. Of these, 601 are general bugs, 110 usability, 42 performance, and only 17 security.

The following tables show the back-end system performance when applying the multiple expert's criteria. The overall accuracy is about 82% compared with 69% accuracy of the baseline system. Similar to the front-end system, the back-end accuracy is improved when we kept the reviews labeled by two or more.

TABLE 5.19: Overall accuracy of back-end system using data labeled with two or more experts

Backend	Accuracy		Precision	Recall	F-Measure
Baseline (all dataset)	No of Correctly Classified Instances with accuracy	Incorrectly Classified Instances			
	376 68.99%	169 31.01%	0.68	0.69	0.68
Using high confidence data	385 77.62%	111 22.38%	0.79	0.78	0.72
Two or more experts	216 82.44%	46 17.56%	0.84	0.82	0.78

TABLE 5.20: Per-Class performance of the backend system when using high confidence data

	General Bug			Security			Performance			Usability		
Baseline(all dataset)	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
	0.84	0.89	0.86	0.08	0.27	0.12	0.12	0.09	0.11	0.36	0.14	0.21
Using high confidence data	0.78	0.99	0.87	1.00	0.08	0.14	1.00	0.11	0.19	0.74	0.24	0.37
Two or more experts	0.82	0.99	0.89	1.00	0.25	0.40	1.00	0.11	0.20	0.90	0.21	0.34

As shown in Table.5.20, the general bug category has the best precision and recall, since the accuracy was about 0.82 for the precision and 0.99 for the recall. Then, the usability was the next better category since the precision was about 0.9 and recall was very low. It is about 0.21 and the performance and security categories were very low even so it has 100% precision but the recall 0.25% for the security and 11% for the performance which is not good, this is appear below in the confusion matrix.

For the sentimental analysis system, 308 reviews meet the two or more expert's criteria. 207 of them are positives, 29 negatives, and 9 neutrals. The sentimental analysis system results when using two or more expert's criteria are shown in the following Table.5.21 and 5.22.

TABLE 5.21: Overall performance of the sentimental analysis system using data labeled with two or more experts

Baseline (all dataset)	**Accuracy**		**Precision**	**Recall**	**F-Measure**
	Correctly Classified Instances with accuracy	**Incorrectly Classified Instances**			
	644 83.85%	124 16.15%	0.841	0.84	0.84
Using High confidence data	624 90.17%	68 9.83%	0.87	0.90	0.88
Two or more experts	90 85.71%	15 14.29%	0.89	0.86	0.87

TABLE 5.22: Per Class performance of the sentimental analysis system using data labeled with two or more experts

	Negative			Positive			Neutral		
Baseline(all dataset)	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
	0.63	0.59	0.61	0.93	0.93	0.93	0.324	0.35	0.34
Using high confidence data	0.85	0.40	0.55	0.91	1.00	0.95	0.01	0.01	0.01
Two or more experts	0.46	0.60	0.52	0.95	0.91	0.93	0.20	0.25	0.22

The overall accuracy is about 86% compared with the accuracy of the baseline, which is 84%. The positive class has the best recognition rate compared with the other two classes.

5.5 Investigate the impact of data size on the system performance

In all of our previous experiments, the dataset was divided into 66% training and 33% testing. As we have seen, in some experiments and after applying filtration criteria, the amount of the resulted data is small in total and very small for some classes. The amount of data for each class has a strong effect on the system performance. In this section, we aim to study the impact of data size on the system performance by varying the amount of training data. This is achieved by applying cross validation scheme, where a percentage of the available data is taken out for testing and the remaining is used for training. Then, other reviews (same percentage) which are not used for testing in the first round are used for testing and the rest for training. This experiment is repeated until every single review is used for training and for testing, but not at the same time.

For this purpose, we used 5-fold and 10-fold cross validation criteria. By this, the amount of training and testing data is varied where the other configurations of the baseline system are kept fixed. The results are presented in Table.5.23.

Using cross-validation technique improves the performance of the three components of our baseline system. We also observe that there is no significant difference between the 5-fold and 10-fold cross-validation.

TABLE 5.23: System performance when using 5-fold and 10-fold cross-validation

	Accuracy	Precision	Recall	F-Measure
Front-end system				
Baseline (66%-34% division)	0.7291	0.74	0.72	0.73
10-fold cross-validation	0.8243	0.823	0.824	0.82
5-fold cross-validation	0.8232	0.82	0.82	0.82
Back-end system				
Baseline (66%-34% division)	0.6899	0.68	0.69	0.68
10-fold cross-validation	0.7937	0.75	0.79	0.75
5-fold cross-validation	0.7875	0.744	0.788	0.738
Sentimental analysis system				
Baseline (66%-34% division)	0.8385	0.841	0.84	0.84
10-fold cross-validation	0.9081	0.88	0.91	0.89
5-fold cross-validation	0.996	0.88	0.91	0.89

5.6 Investigate the impact of class-dependent data balancing on the system performance

As we have seen in all of the previous experiments, the class-dependent data is imbalanced. i.e. the data available for each class (for the frontend, backend and sentimental sub-systems) is not equal. This makes the system biased to the class with more data. In this section, we apply some techniques to overcome this issue, as described in the following sub-sections. All of the experiments presented in this section are conducted with the same classifier (Multi-nominal Naïve Bayes), 10-fold cross-validation, data with high confidence labels and the same baseline configurations.

5.6.1 Data re-sampling

In all of the earlier presented experiments, the per-class available data is imbalanced. Usually, a classifier performs well when the classification technique is applied to a data set evenly distributed among different classes. The imbalanced

class distribution problem occurs when one class is represented by a large number of examples (majority class) while the other is represented by only a few (minority class). In this case, a classifier usually tends to predict that samples have the majority class and completely ignore the minority class. This is known as the class imbalance problem [14].

A popular way to deal with the class imbalance problem is sampling. Sampling methods modify the distributions of the majority and minority class in the training data set to obtain a more balanced number of instances in each class [14]. To minimize class imbalance in training data, there are two basic methods, under-sampling and over-sampling.

Under-Sampling: in order to make data balanced among the classes, this method randomly selects a subset of data from the large set and then remove it, to make balance between classes [2]. Hence, an under-sample approach is aimed to decrease the skewed distribution of majority class and minority class by lowering the size of majority class [2]. Its suitable to use the under-sampling when the data is very large in the majority class and its benefits to reduces the training time and storage [19].

Figure.5.3 illustrates the distribution of samples in a data set before and after applying under-sample approach. For example, from the Figure we find the circle is represented minority class, which has 34 instances. So, for this reason we take randomly only 34 instances from other shapes, which are represented majority classes in this case. The drawback of this technique is that there is no control to remove patterns of the majority class. Thus, it can discard data potentially important for the classification process [2], which degrade classifier performance.

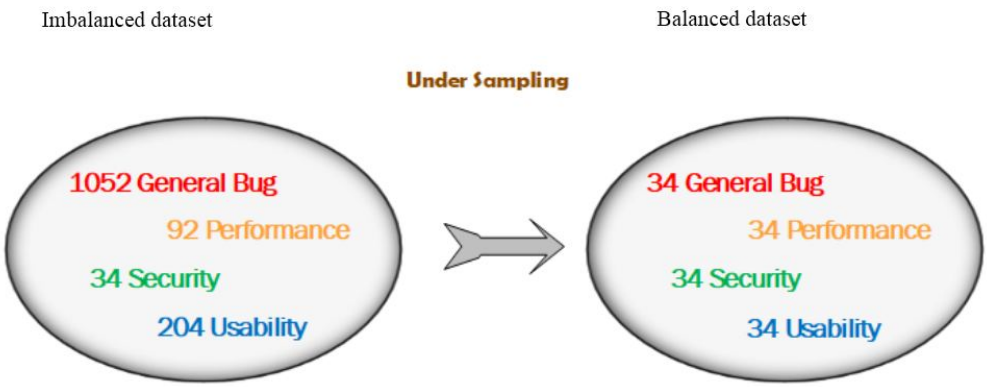

FIGURE 5.3: The Distribution of Samples Before and After Applying Under-Sample Approach

Over-Sampling: in order to make balance in the classes, this method randomly selects data from the lower class and then replicate the selected examples and add them to the data set [19]. It is different from under-sample approach, so there is no loss in the data, and all instances are employed. However, the major problem of this technique leads to a higher computation and more memory storage.

Figure.5.4 illustrates the distribution of samples in a data set before and af-

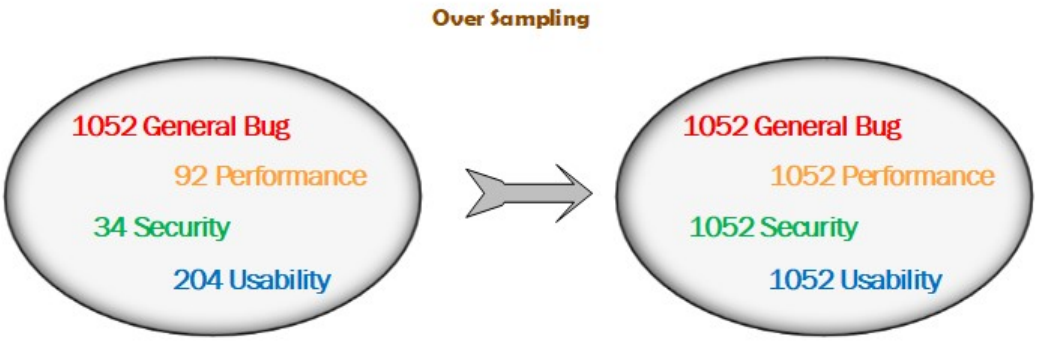

FIGURE 5.4: The Distribution of Samples Before and After Applying Over-Sample Approach.

ter applying over- sample approach. For example, from the Figure, we find the circle represents the majority class, which has 1052 instances. So, for this reason we replicate instances from other shapes, which represent minority classes until they reach to 1052 instances.

Many methods have been proposed to mitigate this scenario, such as to build a new data set in order to reduce the imbalanced data or generate automatics artificial data set. some others proposed techniques that make data balance, based on under-sampling data or over-sampling data. [19].

TABLE 5.24: System performance with applying re-sampling technique

	Accuracy	Precision	Recall	F-Measure
Front-end system				
Baseline (10-fold)	0.8243	0.823	0.824	0.82
+ Resampling using multi nominal naïve bayes	0.8778	0.880	0.878	0.877
Back-end system				
Baseline (10-fold)	0.7937	0.75	0.79	0.75
+ Resampling using Multi nominal naïve bayes	0.882	0.877	0.883	0.870
Sentimental analysis system				
Baseline (10-fold)	0.9081	0.88	0.91	0.89
Resampling using Multi nominal naïve bayes	0.958	0.955	0.957	0.953

In our case, the imbalanced data set solved by using two methods; under-sampling and over-sampling. The first one is by building new instances to get more data for small data instances, where, the second is building an automatic data in order to make the data balanced. There are a lot of techniques exist in the litreture to deal with this problem such as SMOTE, under sampling, over sampling, class balancer and Resample to build data. we chose supervised resampling technique based on the recommendations of the literature reviews [19].

The distribution of the reviews over the defined classes of the front-end system, is as follows, 1382 Bug, 451 new feature and 1865 sentimental. The performance result of the front-end system with applying re-sampling techniques and keeping the other configurations fixed, such as 10-fold cross-validation, classifier type, features, etc.

TABLE 5.25: Per-class performance of the frond-end system with applying re-sampling technique

Frontend	BUG			New Feature			Sentimental		
	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
Baseline (10-fold)	0.79	0.85	0.82	0.67	0.57	0.62	0.89	0.87	0.88
+ Resampling using multi nominal Naïve Bayes	0.824	0.918	0.868	0.838	0.654	0.735	0.931	0.902	0.917

TABLE 5.26: Per-class performance of the Back-end system with applying re-sampling technique

	General Bug			Usability			Performance			Security		
	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
Baseline(10-fold)	0.81	0.98	0.89	0.50	0.06	0.11	0.32	0.1	0.15	0.670	0.28	0.40
+ Resampling using multi nominal niave bayes	0.890	0.988	0.936	0.897	0.642	0.749	0.691	0.413	0.517	0.846	0.324	0.468

As shown in Table.5.24 , the accuracy is increased when we used Resample technique with multinational niave bayes algorithm to 88% for the front end an back end systems , and 96% for the sentimental analysis system.

As shown in Table.5.25 , the precision and recall is increased when we used Resample technique multinational niave bayes for each general bug ,new feature and sentimental

As shown in Table.5.26 , the precision and recall for each general bug, security, usability and performance are increased when we used Resample technique with multinational niave bayes algorithms.

TABLE 5.27: Per Class performance of the sentimental analysis system with applying re-sampling technique

	Positive			Negative			Neutral		
Classification 10 fold Sample	Prec	Rec	F.M	Prec	Rec	F.M	Prec	Rec	F.M
Baseline (10-fold)	00.78	0.46	0.58	0.92	0.99	0.96	0.19	0.04	0.07
+ Resampling using multi nominal niave bayes	0.963	0.997	0.980	0.889	0.620	0.731	0.889	0.500	0.640

As shown in 5.27 , the precision and recall for each positive negative and neutrals are increased when we used Resample technique with multinational niave bayes algorithms.

Its important to mention that once we applied the random forest with re-sampling techniques it gives a great result nearest from the multi nominal naive bayes.

Chapter 6

Conclusion and Future Work

6.1 Conclusions

in this thesis, we presented our proposed system for automatic classification of healthcare apps reviews extracted from Google play store. Around 90,000 reviews were retrieved from 10 common healthcare applications. A subset of this data is manually annotated by software engineering experts indicating their confidence for each annotation. A comparison of commonly used classifiers such as Naïve Bayes, Naïve Bayes Multinomial, Random Forest, and Support Vector Machines for multi-class apps reviews classification is presented in this thesis. Our experiments show that Multi-nominal Naive Bays can classify mobile apps reviews into bugs, new feature, and sentimental with an accuracy of 87%, and into general bug, usability,security and performance with an accuracy of 88%. The best result of the sentimental analysis system is 90%. in addition, the experiments show that the overall performance is improved when we use the data subset with high confidence labels and when two experts agree on the same label. Re-sampling technique was successfully used to overcome the data

imbalance problem in our data set, so the accuracy improved to 89% for the mobile apps reviews into a set of classes; bugs, security, new feature, performance, and usability and 96% for the sentimental reviews .

We have also found that all of the described systems can recognize 'bug' class with the highest accuracy, followed by the 'new feature' and the 'usability' class.

As we illustrated earlier in data annotation, experts were asked to express their confidence level (high, medium and low) for each annotation. An important observation, ,which confirms our expectation, that the system accuracy significantly increased when doing training and evaluation on the subset with high confidence annotations only. Moreover, the accuracy is further increased when only the reviews with at least two annotators agree on the annotation with high confidence.

On the other hand, the system ability to classify the reviews correctly decreases by around 62% by including the reviews with medium and low confidence.

6.2 Future Work Plan

In the future, We would like to study the correlation between the user rating (1-5) and sentimental analysis from one side, and our front-end and back-end systems from the other side.

We believe that some special keywords are good indicators of the reviews category. For example, words such as a crash, solve, etc. indicate that the user is describing a bug in the application. Words like please, add, wish to improve.etc.

indicate to the new feature request, and so on. To exploit such keywords for our classification problem, we will build a dictionary of specially selected keywords for each class and integrate it into our system. The effectiveness of this idea on the overall system accuracy will be studied, as well as its effect on each individual class.

At the system level (i.e. machine learning techniques), we would like to investigate in detail, the effectiveness of the state of the art techniques in the machine learning field, which are based on the Deep Neural Networks (DNN). DNN techniques have been successfully used and outperformed others techniques in different fields in natural language processing, image and speech processing and many others.

Bibliography

[1] M Aitken and J Lyle. "IMS Institute for Healthcare Informatics". In: *Patient adoption of mHealth-report by the IMS Institute for Health care Informatics* (2015).

[2] Aida Ali, Siti Mariyam Shamsuddin, and Anca L Ralescu. "Classification with class imbalance problem: a review". In: *Int J Adv Soft Comput Appl* 7.3 (2015), pp. 176–204.

[3] Eirik Årsand et al. "Mobile phone-based self-management tools for type 2 diabetes: the few touch application". In: *Journal of diabetes science and technology* 4.2 (2010), pp. 328–336.

[4] Maged N Kamel Boulos et al. "How smartphones are changing the face of mobile and participatory healthcare: an overview, with example from eCAALYX". In: *Biomedical engineering online* 10.1 (2011), p. 24.

[5] Maged N Kamel Boulos et al. "Mobile medical and health apps: state of the art, concerns, regulatory control and certification". In: *Online journal of public health informatics* 5.3 (2014), p. 229.

[6] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[7] Chaochang Chiu et al. "APP REVIEW ANALYTICS OF FREE GAMES LISTED ON GOOGLE PLAY". In: ().

[8] Susan Dumais et al. “Inductive learning algorithms and representations for text categorization”. In: *Proceedings of the seventh international conference on Information and knowledge management*. ACM. 1998, pp. 148–155.

[9] Magdalini Eirinaki, Shamita Pisal, and Japinder Singh. “Feature-based opinion mining and ranking”. In: *Journal of Computer and System Sciences* 78.4 (2012), pp. 1175–1184.

[10] Abdeljalil EL ABDOULI, Larbi HASSOUNI, and Houda ANOUN. “Sentiment Analysis of Moroccan Tweets using Naive Bayes Algorithm”. In: ().

[11] *Explore cell phones medical alert devices that allow safety for seniors*. http://www.greatcall.com. Accessed: 2018-01-15.

[12] Jeffrey H Gitomer. *Customer Satisfaction is worthless, Customer loyalty is priceless: How to make customers love you, keep them coming back and tell everyone they know*. Bard Press Austin, TX, 1998.

[13] Shivani Goyal et al. “The systematic design of a behavioural mobile health application for the self-management of type 2 diabetes”. In: *Canadian journal of diabetes* 40.1 (2016), pp. 95–104.

[14] Xinjian Guo et al. “On the class imbalance problem”. In: *Natural Computation, 2008. ICNC'08. Fourth International Conference on*. Vol. 4. IEEE. 2008, pp. 192–201.

[15] Emitza Guzman and Walid Maalej. “How do users like this feature? a fine grained sentiment analysis of app reviews”. In: *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*. IEEE. 2014, pp. 153–162.

[16] Yin-Fu Huang and Shih-Hao Wang. “Movie genre classification using svm with audio and video features”. In: *International Conference on Active Media Technology*. Springer. 2012, pp. 1–10.

[17] Ronda G Hughes. "Tools and strategies for quality improvement and patient safety". In: (2008).

[18] Claudia Iacob and Rachel Harrison. "Retrieving and analyzing mobile apps feature requests from online reviews". In: *Proceedings of the 10th Working Conference on Mining Software Repositories*. IEEE Press. 2013, pp. 41–44.

[19] Michelle Jagelid and Maria Movin. *A Comparison of Resampling Techniques to Handle the Class Imbalance Problem in Machine Learning: Conversion prediction of Spotify Users-A Case Study*. 2017.

[20] Thorsten Joachims. "Text categorization with support vector machines: Learning with many relevant features". In: *European conference on machine learning*. Springer. 1998, pp. 137–142.

[21] Misha Kay, Jonathan Santos, and Marina Takane. "mHealth: New horizons for health through mobile technologies". In: *World Health Organization* 64.7 (2011), pp. 66–71.

[22] Timothy L Keiningham and Terry G Vavra. *The customer delight principle: Exceeding customers' expectations for bottom-line success*. McGraw-Hill, 2001.

[23] Santosh Kumar et al. "Mobile health technology evaluation: the mHealth evidence workshop". In: *American journal of preventive medicine* 45.2 (2013), pp. 228–236.

[24] Kenny R Lienhard and Christine Legner. "Principles in the Design of Mobile Medical Apps: Guidance for Those who Care". In: (2017).

[25] Walid Maalej and Hadeer Nabil. "Bug report, feature request, or simply praise? on automatically classifying app reviews". In: *2015 IEEE 23rd international requirements engineering conference (RE)*. IEEE. 2015, pp. 116–125.

[26] Walid Maalej et al. “Toward data-driven requirements engineering”. In: *IEEE Software* 33.1 (2016), pp. 48–54.

[27] Walaa Medhat, Ahmed Hassan, and Hoda Korashy. “Sentiment analysis algorithms and applications: A survey”. In: *Ain Shams Engineering Journal* 5.4 (2014), pp. 1093–1113.

[28] Edward Alan Miller and Darrell M West. “Where’s the revolution? Digital technology and health care in the internet age”. In: *Journal of Health Politics, Policy and Law* 34.2 (2009), pp. 261–284.

[29] Rebecca Milner and Adrian Furnham. “Measuring customer feedback, response and satisfaction”. In: *Psychology* 8.03 (2017), p. 350.

[30] Suhaib Mujahid et al. “Examining user complaints of wearable apps: a case study on android wear”. In: *Mobile Software Engineering and Systems (MOBILESoft), 2017 IEEE/ACM 4th International Conference on*. IEEE. 2017, pp. 96–99.

[31] Tony Mullen and Nigel Collier. “Sentiment analysis using support vector machines with diverse information sources”. In: *Proceedings of the 2004 conference on empirical methods in natural language processing*. 2004.

[32] Bo Pang and Lillian Lee. “A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts”. In: *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics. 2004, p. 271.

[33] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. “Thumbs up?: sentiment classification using machine learning techniques”. In: *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics. 2002, pp. 79–86.

[34] Bo Pang, Lillian Lee, et al. “Opinion mining and sentiment analysis”. In: *Foundations and Trends® in Information Retrieval* 2.1–2 (2008), pp. 1–135.

[35] Benjamin A Rosser and Christopher Eccleston. “Smartphone applications for pain management”. In: *Journal of telemedicine and telecare* 17.6 (2011), pp. 308–312.

[36] Ayman Sadig et al. “Culture Effect on Requirements Elicitation Practice in Developing Countries”. In: *International Journal of Software Engineering & Applications (IJSEA)* 8.1 (2017), pp. 49–58.

[37] Abinash Tripathy and Santanu Kumar Rath. “Classification of sentiment of reviews using supervised machine learning techniques”. In: *International Journal of Rough Sets and Data Analysis (IJRSDA)* 4.1 (2017), pp. 56–74.

[38] C Lee Ventola. “Mobile devices and apps for health care professionals: uses and benefits”. In: *Pharmacy and Therapeutics* 39.5 (2014), p. 356.

[39] *Weka 3: Data Mining Software in Java*. https://www.cs.waikato.ac.nz/ml/weka. Accessed: 2018-02-01.

[40] Hui Yang and Peng Liang. “Identification and Classification of Requirements from App User Reviews.” In: *SEKE*. 2015, pp. 7–12.

.1 Appendix A

```
connection = pymysql.connect(host='localhost',
                             user='root',
                             password='',
                             db='medical_apps',
                             charset='utf8',
                             cursorclass=pymysql.cursors.DictCursor)
while True:
    try:

        for elem in driver.find_elements_by_class_name('single-review'):


            if elem is None :
                break;
            content = elem.get_attribute('outerHTML')
            soup = BeautifulSoup(content, "html.parser")
            date = soup.find('span',class_='review-date').get_text()
            author = soup.find('span',class_='author-name').get_text()
            rating = soup.find('div',class_='tiny-star')['aria-label'][7:8]
            title = soup.find('span',class_='review-title').get_text()
            txt = soup.find('div',class_='review-body').get_text().replace("Full Review","")[len(title)+1:]

            temp = author+','+date+','+rating+','+txt+','+title
            with connection.cursor() as cursor:

                sql = "INSERT INTO `users_review` (`Author_name`,`Date`,`Rating`,`review`,`app_id`) VALUES (%s,%s,%s,%s,%s)"
                cursor.execute(sql, (author,date,rating,txt,3))

                connection.commit()
```

FIGURE 1: Snapshot of the python code that used to crawling the medical reviews

```
            txt = soup.find('div',class_='review-body').get_text().replace("Full Review","")[len(title)+1:]

            temp = author+','+date+','+rating+','+txt+','+title
            with connection.cursor() as cursor:

                sql = "INSERT INTO `users_review` (`Author_name`,`Date`,`Rating`,`review`,`app_id`) VALUES (%s,%s,%s,%s,%s)"
                cursor.execute(sql, (author,date,rating,txt,3))

                connection.commit()

    except:

        print('exception.....1')

    try:
        WebDriverWait(driver, delay).until(EC.presence_of_element_located((By.ID, 'reviews')))

    except:

        print('eception')
    if driver.find_element_by_xpath('//*[@id="body-content"]/div/div/div[1]/div[2]/div[2]/div[1]/div[4]/button[2]/div[2]/div/di
        break;
    driver.find_element_by_xpath('//*[@id="body-content"]/div/div/div[1]/div[2]/div[2]/div[1]/div[4]/button[2]/div[2]/div/div')

try:
    print('exception.....2')
except:
    print('exception.....3')
driver.close()
```

FIGURE 2: Snapshot of the python code that used to crawling the medical reviews

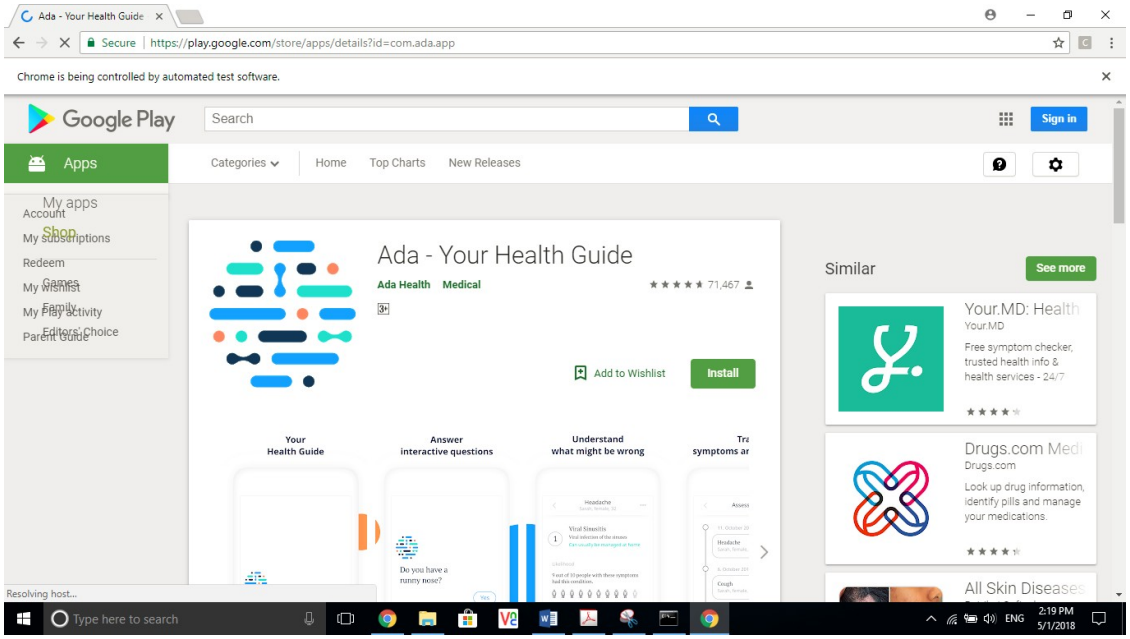

FIGURE 3: Snapshot shows the selenium driver while gathering the reviews

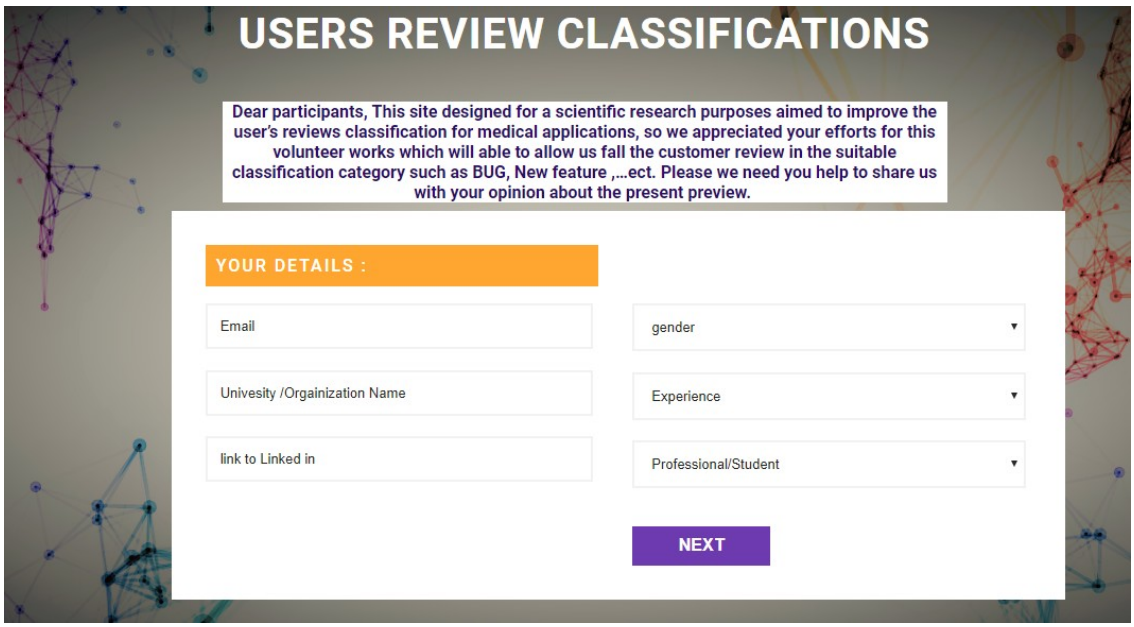

FIGURE 4: snapshot of the application that we developed to the experts in order to classify the reviews

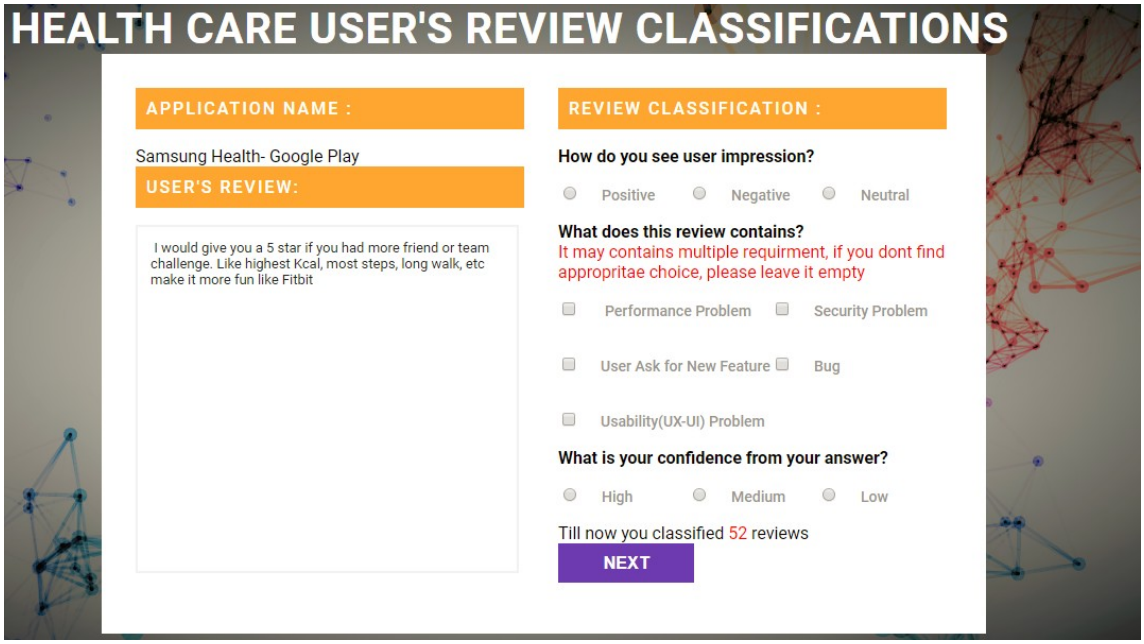

FIGURE 5: snapshot of the application that we developed to the experts in order to classify the reviews

Key	Value
IDFTransform	True
TFTransform	True
attributeindices	First-last
dontCheckCapabilities	False
doNotOperateOnPerClassBasis	False
invertSelection	False
lowerCaseTokens	False
minTermFreq	1
normlizeDoclength	No normalization
periodPruning	-1.0
stemmer	NullStemmer
stopwordsHandler	WordsFromFile –stopwords “C:\\stopwords.txt”
tokenizer	NGramTokenizer –max 2 –min 2 –delimiter “\r\n\t,;()?!”

FIGURE 6: unsupervised attribute StringToWordVector