A Mobile Proximity Job Employment Recommender System

By

Motebang Daniel Mpela

Student Number: 212069926

A research dissertation submitted in fulfillment for the Degree

Magister Technologiae

In

Information Technology

Faculty of Applied and Computer Sciences

VAAL UNIVERSITY OF TECHNOLOGY

Supervisor: Prof: Tranos Zuva

Co-supervisor: Dr. Martin Appiah

December 2020

DECLARATION BY CANDIDATE

"I hereby declare that the dissertation submitted for the degree Magister Technologiae: Information Technology, at Vaal University of Technology, is my own original work and has not previously been submitted to any other institution or higher education. I further declare that all sources cited or quoted are indicated or acknowledged by means of a comprehensive list of references".

MpelaMd________________

Motebang Daniel Mpela

ACKNOWLEDGEMENTS

Foremost, I would like to give thinks to the almighty God for the gift of life, strength and bravery to do my research project. I would also like to thank the Vaal University of Technology through its Research department, Faculty of Applied and Computer Sciences and the ICT department for the opportunity to pursue my studies in MTech in Information Technology.

I would also like to express my sincere gratitude to my supervisor Prof. Tranos Zuva and co-supervisor Dr. Martin Apiah for the continuous support of my MTech study and for their patience, motivation, enthusiasm and immense knowledge. Their guidance helped me in all the time of research and writing of this dissertation. I could not have imagined having better supervisors and mentors for my MTech study.

Last but not the least, I would like to thank my family: my parents Mr. William Mpela and Mrs. Alice Mpela, for giving birth to me at the first place and supporting me spiritually and academically throughout my life.

Abstract

With a rapid growth of internet technologies, many companies have transformed from the old traditional ways of recruiting employees to electronic recruitment (e-recruitment). E-recruiting channels achieved a solid advantage for both employers and job applicants by dropping advertising cost, applying cost as well as hiring time. Job recommender systems aim to help in people – job matching. In this research, a proposed mobile job employment recommender system is a client – server application that uses content – based filtering algorithm to enable the initial selection of a suitable leisure job seeker to a temporary job at a particular place and vice versa. A prototype of a mobile job recommendation application was developed to evaluate the algorithm. The evaluation matrix used to assess the prototype are precision, recall and the F-measure. The precision value was found to be 0.994, the recall value was 0.975 and the F1-score was 0.984. The experimental results of the proposed algorithm show the effectiveness of the system to recommend suitable candidates for jobs at a specified area. The recommender system was able to achieve its main aim of enabling the initial selection of suitable temporary job seekers to a temporary job at a particular place and vice versa. Thus, the results of the proposed algorithm are satisfactory.

Table of Contents

ACRONYM LIST

API	Application Programming Interface
CARS	Context Aware Recommender Systems
CBF	Content Based Filtering
CF	Collaborative Filtering
CV	Curriculum Vitae
EE	Enterprise Edition
FN	False Negative
FP	False Positive
GUI	Graphical User Interface
GPS	Global Positioning System
IDE	Integrated Development Environment
IDF	Inverse Document Frequency
IR	Information Retrieval
LARS	Location Awareness Recommender System
LPS	Local Positioning System
MAS	Mean Absolute Shift
MAE	Mean Absolute Error
MPJERS	Mobile Proximity Job Employment Recommender System
PDA	Personal Digital Assistance
RMAE	Root Mean Absolute Error
SDK	Software Development Kid
TF	Term Frequency
TF-IDF	Term Frequency – Inverse Document Frequency
TN	True Negative
TP	True Positive
VSM	Vector Space Model

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: SCOPE OF THE RESEARCH

1.1 INTRODUCTION

In recent decades, recommender systems have attracted a great deal of attention. Users of recommender systems can take advantage of the search functionality to discover items needed, given the prospect of interacting with a catalogue of millions of items on e-commerce websites, or a system can be used to recommend items that might be relevant to the user. Recommender systems are abused to recommend novels, movies, music videos and millions of different types of products from various e-commerce firms and vendors.

Taking advantage of various user features and user history of interactions with items, recommender systems can use a variety of recommendation techniques to provide relevant items to the user. A recommender system frequently retrains and corrects its model, rather than utilize static weights to merge recommender strategies. A recommender system, for instance, may require a compromise between the context and nature of the recommendation list. Suggesting products for which high importance is calculated by the system can seem like a secure way to ensure solid income. Adomavicius and Kwon (2012) indicate that recommendations often require expansion to give the user more chance to discover interesting items beyond the filter bubble while retaining a reasonable degree of accuracy. This gives flexibility to recommender systems to change trends as well as the ability to deliver customized results for each user.

Recommendation algorithms are also utilized to expand the functionality of mobile systems as well as assisting to work by means of information overwhelming through providing personalized predictions. The predictions can lead to numerous decision -making procedures, like the kind of product to purchase, the kind of book to read or the kind of restaurant to eat at. To determine whether the recommender system is successful, the following must be taken into consideration: filtering technique, system design, and user interface. Recommender systems are not limited to only providing accurate suggestions of items, but also to increase knowledge about the users' preferences and improve user satisfaction (Ricci et al., 2011). Recently, user experience is becoming very vital in online recommender systems (Konstan and Riedl, 2012) even though most researchers put a lot of attention in expanding the systems' accuracy.

However, traditional recommender systems are not fully suitable in a mobile environment. Mobile computing is exceptional in its location-aware ability (Stafford and Gillenson, 2003). Mobile technology adds a relevant but mostly unfamiliar piece of data which is the user's physical location to the recommendation problem. Personalized recommendations in mobile computing provide the chance and task to take physical location into account.

With the wide spread of internet, job employment recommendation has also played a vital part on the online recruiting website and has attracted a lot of research attention. Online employment recommender systems differ from traditional recommender systems in that online employment recommender systems match a similar type of users, for instance a job seeker, to another type of users, for instance employers, while traditional recommender systems suggest objects to users. The intention of recruiting recommendation systems is to suggest a number of available jobs to a job-seeker according to their profiles or to suggest a number of job-seekers to an employer in accordance to the job description.

This research paper analyses the implementation of a mobile proximity recommender system to help in predicting the most suitable candidate for a temporary job employment based on the similarity between the candidates' resumes and the job descriptions and also finding the proximity of recommended resume profiles to the area where the temporary job will be done. Content based recommendation algorithm that uses vector space model is proposed to predict the similarity between text documents resumes and job descriptions and google maps will be used to calculate the distance of recommended job seekers to the area where the job will be done.

1.2 PROBLEM STATEMENT

Tracking applicants and capturing data through information systems have been the main focus of information technology in human resource (HR) departments in the past years. The technology supported internal workflow and interchanging of information between the HR departments and several other departments within an organization (Al-Otaibi and Ykhief, 2012). Rapid growth of technology has reformed ways in which organizations conduct their recruiting processes. Recently, organizations post job advertisements on their online platforms and candidates apply for jobs by creating their profiles on the organization's online platforms. A large number of resumes are received by organizations after advertising a job. Therefore, a large number of online CVs and job descriptions are on the increase.

The main challenge is of recommender systems to filter temporary job seekers who are most suitable for a particular temporary job position and at the same time filter the temporary job position that is relevant to a particular temporary job seeker. This problem becomes even bigger when the recommender system has to recommend a mobile Leisure Temporary Job Seeker

(LTJS); these may be tourists, truck drivers and travelers, for a job position at a particular place using an added characteristic of proximity to the place of work at a specific time.

1.3 RESEARCH QUESTIONS

- How can suitable temporary job seekers be matched to a particular job at a given place and vice versa?
- How can a mobile temporary job recommender system that determines the proximity of the temporary job seekers and the temporary job employer at a particular place be developed?
- How can the effectiveness and efficiency of the mobile proximity job recommender system be measured?

1.4 AIM AND OBJECTIVES

1.4.1 Aim

The aim of this research was to enable the initial selection of suitable temporary job seekers to a temporary job at a particular place and vice versa.

In this study, the research anticipates the task of temporary job identifying recommendation based on the job description, user profile and the physical location, proximity of the job applicant and the area where the job will be done.

1.4.2 Objectives

In order to reach the main objectives, the following are the research objectives:

- To match suitable temporary job seekers to a particular job at a given place and vice versa.
- To develop a prototype of a temporary mobile proximity job employment recommender system that determines the proximity of the temporary job seeker and temporary job employer at a particular place.
- To measure the effectiveness and efficiency of the temporary mobile proximity job recommender system prototype.

1.5 RESEARCH DESIGN

This study followed a qualitative research approach and the proposed research design was literature exploration and an experiment. The datasets of resumes and job description text documents were obtained from Kaggle website. These text documents were then preprocessed using java to remove noise for information retrieval process. The preprocessed text documents were then used in experiments to produce recommendations for suitable jobs. Google maps was also utilized to find the proximity of suitable job seekers to the area where the job will be done.

1.6 CHAPTER SUMMARY

This chapter was aimed to describe the overview of the of the proposed mobile proximity job employment recommender system. First, a brief introduction of the recommender system was described. Then the problem statement was presented. Research questions were then derived as to discover a solution to the problem. The aim and objectives of this research are also demonstrated. Lastly, the research design in was explained.

1.7 THESIS LAYOUT

- Chapter 2, Reviews literature on recommender systems including mobile recommender systems, job recommender systems and also the evaluations of recommender systems.
- Chapter 3, methodology followed in this study and algorithms proposed are discussed.
- Chapter 4, presented the implementation and results of the system prototype obtained in this study.
- Chapter 5, evaluations of experimental results were done.
- Chapter 6, Conclusion and Future Work are discussed.

CHAPTER 2: LITERATURE REVIEW

2.1 INTRODUCTION

This chapter presents the literature review to acknowledge existing theories and hypothesis in the field of recommender systems. Literature review gives a wide-range summary on the topic through previous researches (Mustafa et al., 2017). The literature review in this research study focuses on developing broader arguments and ideas on a mobile proximity job employment recommender system. It serves as the foundation and support structure to the topic.

2.2 BACKGROUND OF RECOMMENDER SYSTEMS

In the beginning, the recommender systems did not create a separate research area and their origins can be drawn back to the cognitive science, information retrieval, forecasting theories, approximation theory and management sciences (Adomavicius et al., 2005). However, the growth of the internet and the rise of e-commerce solutions caused the development of the online recommender systems and since the mid-1990s they have become an important research domain (Adomavicius et al., 2005). The reason for that was the opportunity to share the opinion among a vast number of people who use the internet.

The first known project in the recommendation area was the GroupLens (Rield et al., 2006). The roots of that project can be traced back to the year 1992 when the major objective of the system was to discover computerized collaborative filtering. Thereafter, Usenet news applied collaborative technique to filter information (Montaner et al., 2003). One of the first application to provide a computerized music recommendation was the Ringo Agent (Shardanand,1994), which became available in July 1994. In that application, musical ratings were given to the user, based on their user profile, which changed over time (Shardanand and Maes,1995). The profile allows recommendations to be created by using social filtering methods. This method can be viewed as an automation the word of mouth recommendation. Firefly's system was the program that used the Ringo system. Yahoo also signed up to use it, and further improved this technology (Kangas, 2002). Finally, the book dealer Amazon.com launched the BookMatcher program, using this tool. The BookMatcher was initially used for book recommendations, but later the program started recommending other types of products, often using other recommendation methods (Kangas, 2002).

Today recommender systems are one of the well-established artificial intelligence applications in modern computer science and are being used as typical software components in e-

Commerce, m-Commerce (Sadeh, 2002), social media and tourism systems (Werthner et al., 1999) to recommend magazine articles, books, goods, etc. The most popular ones are movies, music, news, books, research articles, search queries, social tags, and products in general. There are also recommender systems for experts, jokes, restaurants, financial services (Alexander et al., 2007) life insurance, and persons (online dating).

Although RS has been investigated and developed for many years they are still in the area of interest for many researchers. Moreover, there are still many challenges in this area. Each of the existing recommender methods suffers from shortcomings. The goal of this research is to anticipate the task of temporary job recommendation based on the job description, user profile and the physical proximity of the job applicant to the area where the job will be done using content-based filtering.

2.3 RECOMMENDER SYSTEMS ALGORITHMS

The previous section discusses the background behind the use of recommender systems. In this section, different methods and techniques to design and implement recommender systems are discussed. There are several techniques for recommender systems but in this research study only five popular approaches are discussed: collaborative, content-based, demographic, knowledge based and hybrid.

2.3.1 Collaborative filtering

It discovers users that have similar preferences with a group of users called neighborhood and suggest predictions based on what the neighborhood users rated before. Thus, it is also called a user-to-user correlation method. A user does not necessarily get the recommendation of the items that he/she rated previously, but gets recommendation of items that were rated by the neighborhood users. The major step is calculating the similarity between users.

Collaborative filtering (CF) recommendation algorithm can be divided into memory-based and model-based (Mustafa et al., 2017) as show in Figure 2.1. Memory-based CF algorithm generates predictions by accessing the database directly. Due to accessing the database directly, memory-based CF is adaptive to data changes, but requires large computational time to the data size. On the other side, model-based CF algorithm makes use the transaction data to create a model that can generate recommendation (Bobadilla et al., 2013). As for model-based CF, it has a constant computing time regardless of the size of data but it is not adaptive to changes.

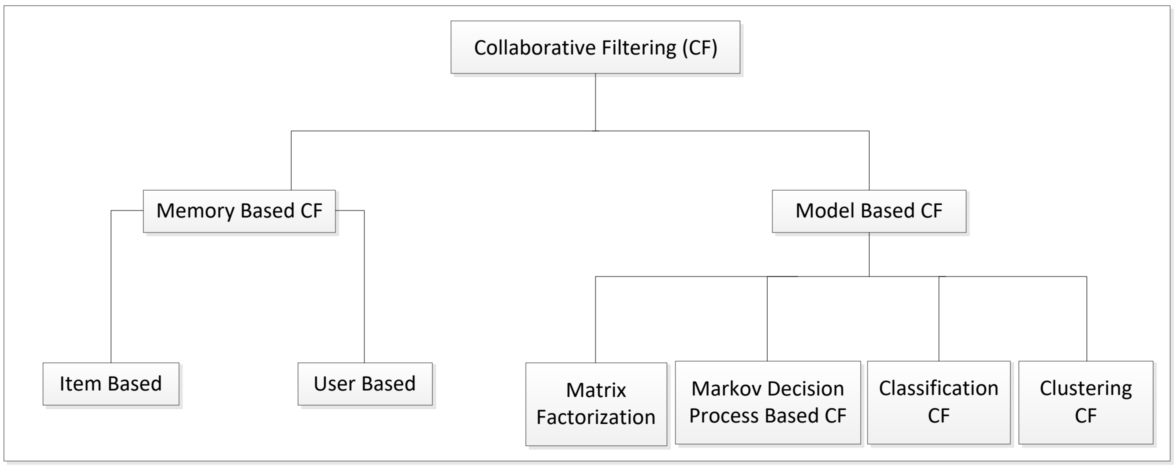


Fig 2.1 - Collaborative filtering techniques (Mustafa et al., 2017)

2.3.2 Content-based filtering (CBF)

The recommendation is based on suggesting items that are similar to items that the user has previously liked. CBF matches the content object of interest to foresee its significance based on the user's profile. Pazzani (2000) emphasized that CBF algorithms generate recommendations by examining the description of the objects that have been previously ranked by the user and the description of objects to be recommended.

Furthermore, CBF suggests objects that have alike content to ones the target user desire. When recommending job seekers and jobs, the personal data and their job desires will the content for job seekers, while job description posted by employers and background description of a company will be the content for jobs. The concept of CBF is choosing similar feature content and compering them by computing the similarity between job seeker and jobs. The expected results will be a list of recommended job seekers and the job positions arranged in order by the similarity index. In essence, the major components of CBF are feature selection and similarity calculation. It is important to consider the influence of common features on the prediction to the target user's preferences or the precise investigation in the job recommender market. The selected features must be represented in a suitable arrangement, like vector space model and the similarity can be computed.

In CBF, items are recommended to users based on their features. Taking a job recommendation, the features will include the following as shown in Table 1.

Table 2.1 – Job Features

	Features
Job Seekers	ID number, Gender, Qualification, Skills, Location
Job	Job title, job description, Salary range, Location

If items can be represented by the same set of attributes with the known set of values associated to these features, then those items are said to have a structured data. When an item has a structured data Machine Learning algorithm can be employed to learn a user profile (Brusilovsky, et al., 2007).

According to Lops, et al. (2010), items' features are textual features from either web pages, emails, product descriptions or new articles. This type of data is said to be unstructured because there are no common features with known values. This poses a challenge in trying to learn the users' profile due to ambiguity of natural language. The major problem is that the semantics of user interest is not captured by the tradition keyword-based profiles because the rely primarily on string matching operation. A match is assumed if some morphological variant or a string is found in both the profile and the document, which is the reason for why the document will be considered relevant. However, this string matching is much challenged by polysemy (the term or word with different meaning) and synonymy (different terms having the same meaning). This can result in relevant information being missed out if the profile does not use the same keywords in the document or different words could be matched due to polysemy resulting in wrong documents being deemed relevant.

According to Lops, et al. (2010), the majority of CBF systems utilize simple retrieval models such as vector space model (VSM) or keyword matching model (KMM) with Term Frequency-Inverse Document Frequency (TF-IDF) weighting. VSM represents text documents spatially. The model represents each document by a vector in an n-dimensional space and each term in a given document's vocabulary corresponds to each dimension of the space. This can be illustrated formally by assuming a term weight vector to represent every document and degree of association between the term and document to be indicated by each weight. If D = $\{d_1, d_2, \ldots, d_n\}$ is set to denote the documents' set and $T = \{t_1, t_2, \ldots, t_n\}$ is the set of words in the document, tokenization, stemming or stopwords removal can be utilized to obtain T (Ricardo and Berthier, 1999). A vector in n-dimensional vector space represents each documents d_j such that $d_j = \{w_{1j}, w_{2j}, \dots, w_{nj}\}$ and document d_j's weight for term t_k is represented by w_{kj}. According to Salton (1989), the most commonly used TF-IDF is based on the empirical observations regarding text include the following assumptions:

- The IDF assumption is that rare terms are as relevant as frequent terms.
- The TF assumption is that multiple occurrences of a term in a document are as relevant as single occurrences.
- Normalization assumption is that the preference between long documents and short documents is the same.

This implies that, terms that occur frequently in one document but rarely in the rest of the corpus are likely to be more relevant to the document's topic. The likelihood of longer documents having a chance to be retrieved most of the time is minimized by normalization of resulting weight vectors. The TF-IDF to underpin the assumption is shown in equation 1:

$$TF - IDF\left(t_k, d_j\right) \cdot log\,\frac{N}{n_k} \qquad (1)$$

where N is the number of documents in the corpus, and n_k is the number of documents in the collection in which the term t_k occurs at least once:

$$TF(t_k, d_t) = \frac{f_{k,j}}{max_k f_{k,j}} \qquad (2)$$

where the frequencies $f_{k,j}$ of the document d_j's terms t_k are used to compute the maximum, the value of the TF can be obtained by utilizing equation 2.

The cosine normalization is utilized to normalize the weight obtained. Equation 3 ensures that weight fall within [0,1] interval and also to ensure that equal length vectors represent the documents.

$$w_{k,j} = \frac{TF - IDF(t_k, d_j)}{\sqrt{\sum_{s=1}^{|T|} TF - IDF(t_k, d_j)^2}} \qquad (3)$$

A similarity measure utilized to compute how close a document is to the query, it can either be the cosine similarity measure or Pearson correlation which are discussed later on in this study.

Generally, content-based recommender systems that rely on VSM, items and user profile are represented as weighted term vector.

2.3.2.1 Strengths of Content – Based Filtering Technique

According to Keenan (2019), CBF technique does have a number of benefits including that of results tend to be highly relevant because content-based recommendations rely on characteristics of objects themselves, they are likely to be highly relevant to a user's interest. This makes them especially valuable for organizations with massive libraries of a single type of content (think subscription and streaming media services). Again, recommendations are transparent as the process by which any recommendation is generated can be made transparent, which may increase users' trust in their recommendations or allow them to tweak them. With collaborative filtering, the process is more of black box; the algorithm and users alike may not really understand why they are seeing these recommendations. In addition, users can get started more quickly because CBF avoids the cold – start problem that often bedevils collaborative – filtering techniques. While the system still needs some initial inputs from users to start making recommendations, the quality of those early recommendations is likely to be much higher than with a system that only becomes robust after millions of data points have been added and correlated. Moreover, new items can be recommended immediately. Related to cold-started problem, another issue with collaborated-filtering is that new objects added to the library will have few (if any) interactions, which means they won't be recommended very often. Unlike collaborative filtering systems, content-based recommenders don't require other users to interact with an object before if starts recommending it. Finally, CBF is technically easier to implement. Compared to the sophisticated math involved in building a collaborative-filtering system, the data science behind a content-based system is relatively straightforward. The real work, as we have seen is in assigning the attributes in the first place.

2.3.2.2 Challenges faces by Content-Based Filtering Technique

On the other hand, Tuan (2019), states that CBF has some flaws in that it has limited content analysis: if the content does not contain information to discriminate the items precisely, the recommendation will not be precise in the end. Secondly, CBF has over-specialization problem: content-based methods provide a limit degree of novelty, since it has to match up the features of profile and items. A totally perfect CBF may suggest nothing surprising. Finally, the new user problem: when there is not enough information to build a solid profile for a user, the recommendation cannot be provided correctly.

2.3.3 Demographic recommender systems

This type of system recommends items based on the demographic profile of the user. The assumption is that different recommendations should be generated for different demographic niches. An example of demographic recommender at work could be the display of ads to users depending on the country they are accessing the system or the language they are speaking.

2.3.4 Knowledge based recommender system

Knowledge-based systems recommend items based on specific domain knowledge about how certain item features meet users' needs and preferences and, ultimately, how the item is useful for the user. In such systems a similarity function estimates how much the user needs (problem description) match the recommendations (solutions of the problem).

2.3.5 Hybrid recommender systems

In the Hybrid Approach, all recommendation approaches that are mentioned above have characteristics and challenges. To get better performance and overcome challenges, these approaches have been combined. In general, collaborative filtering is integrated with other techniques in an attempt to avoid these challenges (Burke, 2002). Burke (2007) presented different ways to integrate collaborative filtering, content-based filtering and knowledge-based approaches into a hybrid recommender system.

2.4 MOBILE RECOMMENDER SYSTEMS

Recommender systems were initially developed for personal computers only. These days, mobile devices are gradually replacing personal computers in many ways and are used for decision making and information. Unfortunately, this task is becoming more difficult every time as the amount of information on the mobile devices keeps increasing. The recommender systems that were generated for personal computers cannot be used for the same purpose because of the many differences in the domain of mobile phones that would make such recommendations less successful.

Some of the main differences and mobile service characteristic are:

- User mobility: This refers to the fact that the user can access a mobile information system in different locations. For instance, a traveller at OR Tambo airport can use a mobile phone to access Airbnb application, to be recommended a room to book around Johannesburg. In this example, the traveller can access the same application everywhere through a mobile device, that is, before or during the travel. An acceptable recommender system should be designed in a manner that the user mobility is retained and exploited by using the knowledge about the current location.

- Device portability: This refers to the physical structure of mobile devices, i.e., limited screen size, limited computation power and data storage. For instance, laptops or smart phones, or Personal Digital Assistants (PDAs).
- Wireless connectivity: This refers to the fact that the devices used to access the recommender systems are networked by means of a wireless technologies such as WiFi. The network is used to access some components of the recommender system that are not residing on the device, and without the need of any wire.

Mobile recommender systems development has been driven but also constrained by these three characteristics. The system can be aware of them and adapt its recommendations to them or try to avoid more complicated scenarios and ignore context data. For example, the above-described Airbnb recommender system could exploit current time and location information of the user to decide on the number of recommended items and their ranking. Or even more, the user interface could be adapted to the device by taking the information of the device type. All of that is in order to improve the usability and quality of the recommender system's service. On the other side it could provide static recommendations just taking explicitly stated user preferences into consideration.

One of the biggest challenges of recommending on a mobile phone is a smaller screen size compared to traditional personal computers. The screen size of future devices is unlikely to improve as it is a necessity for the device's mobility (Davidsson, 2010). The common form to display recommendations is via a list of ranked items in descending order, much like the results of a regular search. Several techniques have been proposed to solve the problem of the small screen when displaying search results on mobile devices, with the typical approach being to display a short description for each item in the result (Davidsson, 2010). A short description of an item is not sufficient so it is necessary to enable a way of displaying more information about it.

Mobile devices also have some advantages so they are not all about limitations. For example, context information can significantly improve the quality of recommendations. Physical location and weather data can be an important and valuable source of information. This type of data discovers the context in which the mobile phone is used at a certain time.

Mobile devices are gradually increasing in popularity with prices of these devices are gradually decreasing, and people affording to own them (Smith and Brown, 2005). It is clear from Polatidis and Geargiadis (2013) that the use of cellular phones as well as the rapid growth of the internet has generated an information overload problem. However, in a mobile recommender system space the usage of mobile devices brings about a new spectrum and changes the focus of recommendations to individuals, thus making personalization a key factor of these mobile devices. It is evident from Woerndl et al. (2007) that in a mobile situation, information personalization is more imperative, for the reason that the confines of mobile devices regarding displays and bandwidth etc. However, it can be said that personalization is not a new research topic within the recommender systems theory because Amazon.com provides a personalized web page to each individual user (Polatidis and Geargiadis, 2013).

Mobile recommender systems increase loyalty by taking personal information as input from the user and generating recommendations locally or in a distributed environment and it directs the predictions in a form of a recommendation to the interface that the user is using. With the hasty development of mobile computing technologies, various kinds of mobile applications gained popularity (Gavalas and Economou, 2011). As a groundbreaking technology, mobile computing enables the access to information anytime, anywhere, even in surroundings with scarce physical network connections. Amid others, the operative use of mobile technology in the field of mobile tourism has been actively studied. Along this line, mobile recommender systems (i.e. recommender systems tailored to the needs of mobile device users) represent a relatively recent thread of research with numerous potential application fields (e.g., mobile shopping, advertising/marketing and content provisioning) (Ricci, 2011). For instance, Yang et al. proposed a location-aware recommender system that accommodates customers' shopping needs with location-dependent vendor offers and promotions (Yang et al., 2008). Yuan et al (2010) introduced a framework which enables the creation of tailor-made campaigns targeting users according to their location, needs and devices' profile.

Due to the adoption of mobile devices, several studies have been done in the mobile recommender systems space. Yang et al. (2008), in their research "A location-aware recommender system for mobile shopping environments" study recommender systems that consider the location of the mobile user and proposes a location-aware recommender system that considers a user's shopping needs with a location-dependent vendor's offers and they had satisfactory results greater than 70%. However, Dunlop et al. (2004), Setten et al. (2004) and Tung and Soo (2004) state that only a few recommender systems are designed for mobile users and, very few existing mobile recommender systems are conversational. Ricci and Nguyen (2005) illustrated a mobile recommender system called MobyRek which has been designed to run on a mobile phone with limited input from the user thus reducing user interaction and making the system proactive. Proactivity for recommender systems is a good utility.

Baltrunas et al. (2012) further argue that context is crucial in mobile recommender systems. In their research of Context Aware Recommender Systems (CARS), they study the relationship between an item rating and context for example an outdoor restaurant can be rated 5 in summer because it is hot then be given a lower rating in winter because customers feel cold when visiting the restaurant.

Polatidis and Georgiadis (2013) argue that granting that recommender systems are in every personal computer and mobile device currently, there are many factors that manipulators of mobile devices take into consideration and avoid their use. These factors are tightly cohesive to privacy. Additionally, the drift in the internet era is social networks and its offshoots that amount to huge sums of data being swapped daily. These data should be in recommender systems use to improve personalization.

2.5 LOCATION AWARENESS RECOMMENDER SYSTEMS (LARS)

With an increasing amount of internet usage recently, people of different or same cultures, professions, age groups, genders and locations connect with each other to create various kinds of relations like friendship, cooperative information and common interest (Marin & Wellman, 2011). Different internet users make use of various internet facilities that are the computerized delineation of real-world networks. The utilization of internet facilities advances such administrations as well as enable the development of system by giving web users to share thoughts, interests, occasions, area, and exercises to fortify the connection with one another. The commencement of imparting areas to one another helps fortify the relationship between genuine systems and online interpersonal interaction administrations (Zheng et al., 2012). LARS is the framework where individuals share the area inserted data with one another (Symeonidis et al., 2014). Additionally, users in LARS share area labeled media content, for example, text, recordings and photographs (Zheng et al., 2012). Besides, the physical area includes the prompt area of a user with a timestamp just as the area history of the given user for a specific period. At the point when same area is shared by at least two users, the data additionally incorporates the total information on the users' basic conduct, interests, and exercises removed from the users' area history and area labeled data (Bao et al., 2013).

2.5.1 Facilities offered by LARS

Existing LARS facilities can be classified into geo labeled media based, point location based, and trajectory-based.

2.5.1.1 Geo Labeled Media Based

Geo-labeled media-based administrations permit clients to include area with users’ media substance, for example, text, recordings and photographs that were made in the physical world (Majid et al., 2013). Aloof labeling happens when a user expressly makes and includes the content alongside the area. Geo-labeled media-based administrations permit a user to see different users’ content in a topographical setting by utilizing advanced guides on personal digital assistance (PDAs) (Chon and Cha, 2011). Well known applications that give LARS administrations incorporate Panoramio3, Geo Twitter1 and Flickr2. The expansion of just area measurements, (for example, latitude, longitude) does not really pull in users, as they are more intrigued by real media content (Ye et al., 2010). In this manner, the option of area data just goes about as an extra to advance and bind together the media substance. Ye et al. (2010) further showed that expansion of the area highlight does not have a lot of effect on the associations and connections among the users; rather it is the media content that is answerable for such associations and connections amongst users.

2.5.1.2 Point Location Based

Point location-based administrations permit users to include and share users' areas, for example, cafes, shopping centers, or films (Gao et al., 2015). Well-known applications that offer such administrations include Facebook, Instagram, Twitter and Foursquare that urge users to share their current area. Users of these applications are given alternatives to do a registration at various areas that are visited by users in their every day schedule to share encounters and information by issuing a feedback (Sarwat et al., 2013; Ye et al., 2010). For instance, a user can share their perspectives about a supper to their locale on an online social webpage while utilizing their PDAs. In addition, these administrations additionally monitor the user's geospatial registration information, (for instance, time and latitude/longitude coordinates) (Ye et al., 2010). In Foursquare, in the wake of checking in at various areas, the application identifies and focuses the user. The user that has the best number of check ins at a specific area has been topped as Mayor. One of the primary points of interest of such administrations that permit constant area tracing of users is that they can find companions around their physical areas that help in boosting a user's social exercises in the physical world. For example, subsequent to finding a companion's physical area from his/her interpersonal organization, one can offer the companion to have a lunch or shopping action. The utilization of the feedback in location-based administrations permits clients to share remarks and proposals that can be either positive or negative. Such tips or inputs are essential in amassing proposals. Not at all like geo labeled media, a point area (setting) is the primary part for the clients that the association between the clients and the media substance, for example, inputs, identifications, and tips are related with the point area (Zheng et al., 2012).

2.5.1.3 Direction Based

Direction based facilities permit users to include both guide areas and the courses toward that point area. The most widely recognized applications that offer these facilities are SportsDo6, Bikely4, and Microsoft GeoLife5. Direction based methodologies register data by removing information about a user’s check in patters at various areas, term of stays, and the ways used. These facilities permit users to include data, for instance, separation, speed, course, and term about a particular direction, just as the user’s media content, for example, labels, tips, and photographs alongside the given direction (Ye et al., 2010; Ying et al., 2010). Different users of a similar network can take direction from the encounters of their companions by following the direction utilizing advanced guides or PDAs. In rundown, such facilities offer both where and when along how and what.

2.5.1.4 Distinguishing Features of Locations

In LARS, the focal point of the suggestions stays on the areas close to the media contents. In this manner, it is significant for the recommender system to perceive and consider one-of-a-kind features of the areas to activities to a user that meets the standards of both precision and nature of suggestion.

2.5.1.5 Location Hierarchy

There are numerous scales by which area can be thought of. An area can be a little shop or cafe or it very well may be a major city or town. These areas, little or greater structure a progressive system where areas at the base are allude to smaller geological zones (Xiao et al., 2014). For example, an area (eatery, film, shopping center) may have a place with a network, a network has a place with a town, and a town has a place with a nation, etc. The various degrees of progressive detailing of areas lead to assorted user area and area diagrams. Bao et al. (2013)

proposed that regardless of whether a user has indistinguishable area chronicles, diverse user area and area diagrams will be planned. The significance of the various leveled connections and their thought is basic for the recommender system in light of the fact that these connections have a noteworthy job in setting up the associations amongst users (Wang et al., 2013). For instance, users that share areas like eatery or shopping centers that are considered as lower-level things in the progressive system conceivably have solid associations than clients who share areas like, towns or nations that are considered as more significant level pecking order. As an outcome, the progressive property in LARS is one of a kind and should be considered.

2.5.1.6 Distance of locations and users

The second distinctive property of areas in LARS is distance. To discover the quality of affiliation and associations between users, distance should be thought of. The shorter the distance the more grounded will be the association. There are three geospatial distance relations characterized to figure the affiliation and associations among users utilizing distance (Wang et al., 2013). The three geospatial distance relationships are the separation between the users, the separation between an area and a client, and the separation between two areas. Separation in each of the three previously mentioned cases can influence the recommender systems in three potential manners. In the principal case, the separation between two users shows the likeness between users. For example, users who have comparable history of visiting same area have high need to have comparative inclinations and intrigue (Arora et al., 2014; Xiao et al., 2010) and the users who have comparative local location are perhaps companions (DeScioli et al., 2011). In the subsequent case, the separation among clients and areas shows the likelihood of a client's fascination in the particular area. For instance, users every now and again visit close by places than different spots that are far away from users' homes (Levandoski et al., 2012). In the last case, the separation between two areas shows the relationship between various areas. For example, cafes and shopping centers are generally arranged close to one another (Ye et al., 2011).

2.5.1.7 Sequential Ordering

Most users visit most loved spots at customary stretches. These customary stretches make a connection between users in a consecutive request. For instance, two users of a similar organization normally feast at two distinct cafes and meet again in the film, a requesting of visits can be made that may give some regular inclinations among them (Bao et al., 2013) or that may conceivably show traffic conditions (Tang et al., 2010). Figure 4 shows the visit designs clients can have in a LARS. A user can check in various areas and include these areas with media contents like feedbacks, labels, and tips.

2.6 JOB EMPLOYMENT RECOMMENDER SYSTEMS

Recent literature show high demands of information technology (IT) for human resource sector and hiring procedures. Many organizations have set attention on using online – recruitment platforms as the main recruiting channel. Job advertisements are automatically published on online platforms immediately they are captured into the system. Job seekers on the other sites create their profiles to apply for available jobs on the online platform. The profiles of job seekers are stored into the system, allowing job seekers to use them again in the near future. The other functionality allows the organization the chance to generate the candidate pool. That is, the organization can view all job seekers' profiles and recommend suitable job. Suitable job seekers' resumes are taken to HR department to process. Furthermore, the system keeps track of the job seekers' statuses throughout the application process and communicate required processes (Malinowski et al., 2005).

In many cases, online recruitment channels use the recommendation algorithm and Boolean search that does not solve the difficulty of a job-candidate matching as conditional structure sufficiently (Malinowski et al., 2006). Several studies have included the idea of recommender systems into the recruiting domain. Some researchers suggest that single features such as personality, personal knowledges and intellectual capabilities which regulate the matching amongst individuals and the task to be accomplished must be considered (Malinowski et al., 2008). Also, the upcoming team members and the interactive features which determine the matching amongst individuals must be considered. In this case, literatures differentiate amongst candidate-team, job-candidate, and candidate-company fit (Sekiguchi, 2004). Therefore, the hiring technique should include all the features. Shifting filtering techniques to find candidate remains problematic, yet still an achievable objective. Thus, most filtering techniques used to recommend jobs and people to resolve problems of online hiring channels (Laumer and Eckhardt, 2009).

The idea of CBF involves matching items with the same content information to one of the target users. To match jobs to candidate, job preferences and user credentials act as the content information for job applicants. To match job applicants to jobs, the content information of employers are job description and the background of the company. The common concept of CBF is obtaining the content information of job description and job seekers and find the similarity. The key components of CBF include selection of features and computation of similarity. For selection of features, similar features have to be selected in accordance with the scientific analysis in the job recruiting market (Zheng et al., 2012) or according to the target users' preferences. Features that are selected must be characterized in a suitable vector form (vector space model) so that the similarity can be computed.

2.6.1 System requirements for job employment recommendation

According to Keim (2007), there are significant requirements introduced in research works that ought to be inferred when suggesting candidates for a particular job employment and they include:

1. The matching of individuals to jobs depends on skills and abilities that individuals should have.
2. Recommending people is a bidirectional process that needs to take into account the preferences not only of the recruiter but also of the candidate.
3. Recommendations should be based on the candidate attributes, as well as the relational aspects that determine the fit between the person and the team members with whom the person will be collaborated.
4. Individual is considered to be unique; we cannot choose a single person several times such as a movie or book.

Recruitment recommendation issue is bidirectional proposal between candidates and employment. The recommendation procedure can be partitioned into two sections: candidate recommendation and employment recommendation. The structure knowledge of these two sections is the equivalent generally (Malinowski et al., 2006; Yu et al., 2011). For a candidate, the job with higher coordinating degree ought to be prescribed. Likewise, for an employment, the candidate with higher coordinating degree ought to be prescribed (Yu et al., 2011). In essence, matching items are either the highest job seekers that match the description of employment in place or highest job profiles that match the job seekers' preferences. Furthermore, Fazel-Zarandi and Fox (2010) referenced that aptitudes requirements coordinating need to differentiate between "must have" and "pleasant to-have" prerequisites in the coordinating procedure. Must-have requirements are imperatives that ought to be controlled by the candidate, while pleasant to-have prerequisites are preferences that are mulled over when positioning candidates.

2.6.2 Job recommendation information

Job seekers and jobs ought to be coordinated dependent on specific standards that are utilized as markers of execution on the job. In choice hypothesis, the accessible data at a specific season of choice is called indicator information which includes the individual characteristics. The real determination strategy is called indicator. The expectation procedure is alluded to the appraisal of the measures utilizing the indicator information and a strategy explicit method of information blend (Farber et al., 2003). However, building job seeker profiles, the meta-data are separated from existing resumes. Farber et al. (2003) proposed a framework that constructs user profile in enlistment condition legitimately from investigating the practices of web user. In this framework, client profiles are built by inactively recognizing the snap stream and read-time conduct of users. Malinowski et al. (2006) utilized user profile information for their CV-recommender: segment information, instructive information, professional training, language abilities and IT aptitudes, grants, distributions, others. As a rule, job seeker's profile is made out of three segments.

1. Personal information about the employee, such as the first name, last name, and location.

2. Information about the current and past professional positions held by the candidate. This section may contain company names, positions, company descriptions, job start dates, and job finish dates. The company description field may further contain information about the company (e.g., the number of employees and industry).

3. Information about educational experiences, such as university names, degrees, fields of education, start and finish dates (Paparrizos et al., 2011).

Malinowski et al. (2006) suggested coordinated effort measures, where job applicants were approached to rate the activity profiles utilizing 5 focuses scale extending from 1 to 5. Applicants were approached to assess whether the profiles intrigued to them or not concerning their profession point of view. As of that meta-data, various features were extricated to prepare and test proposal (Paparrizos et al., 2011). Then again, the job profile ought to be built to depict the requirements as well as posting of every single significant expertise that a representative for the job ought to have (Laumer and Eckhardt, 2009). Additionally, the nature of the proposal framework can be evaluated utilizing factual exactness measurements, for instance, the Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) or Correlation figurings (Herlocker et al.,1999; Malinowski et al., 2006; Su and Khoshgoftaar, 2009).

2.7 TEXT CLUSTERING ALGORITHMS

The text-based document clustering algorithms characterize each document according to its content, i.e. the words (or sometimes phrases) contained in it. The basic idea is that if two or more documents contain many common words, then it is likely that these documents are very similar. Fig 2.2 shows an example of document clustering where similar objects are clustered together.

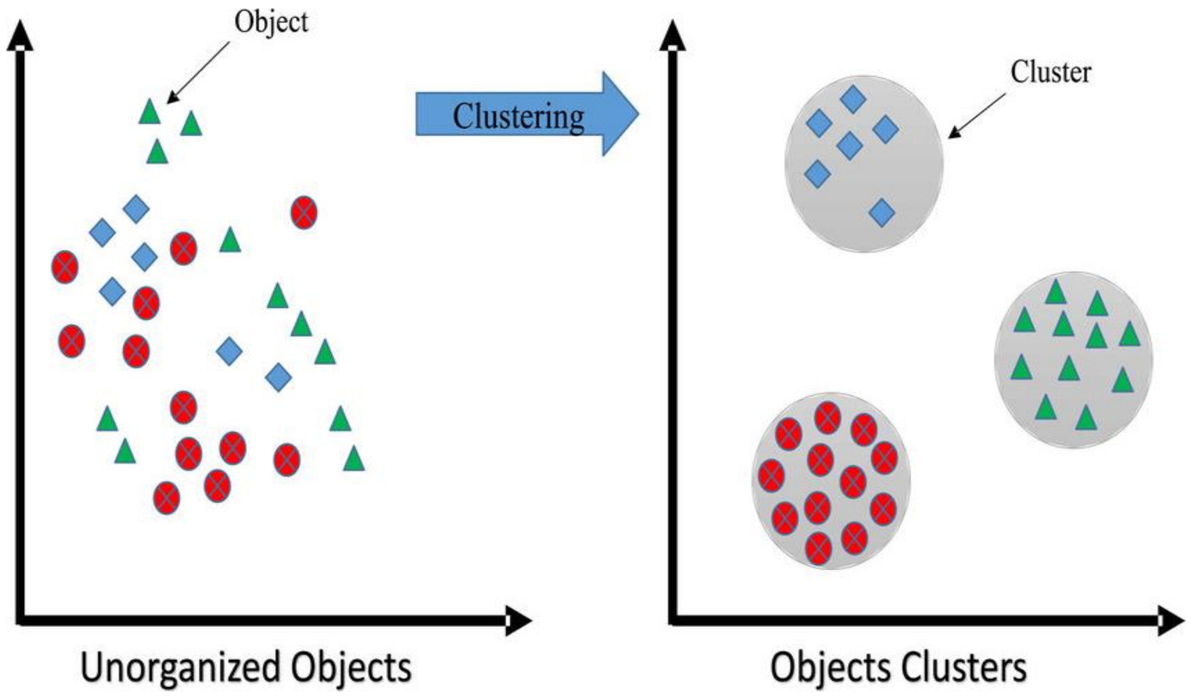


Fig 2.2 - An example of document Clustering (Abualigah et al., 2018)

The similarity of the documents with the existing clusters is computed using distance functions and similarity metrics. In their majority traditional methods require an a-priori knowledge of the number of clusters in order to decide the target cluster of each new document. Algorithms such as K-means and EM aim to minimize the within-cluster distance (Berkhin, 2002). Each cluster is represented by the mean (or weighted average) of its points, the so-called centroid and euclidean or Hellinger (Brants et al., 2003) distance is usually computed. Other approaches, such as the threshold-based algorithm use similarity metrics. Many similarity metrics have been proposed (Brants et al., 2003) with the most popular among them, the cosine similarity (Kumaran and Allan 2004). The similarity between document d and cluster c can be calculated as in equation 4:

$$sim(d,c) = \frac{\sum_w weight(w,d) * weight(w,c)}{\sqrt{\sum_w weight(w,d)^2} * \sqrt{\sum_w weight(w,c)^2}} \tag{4}$$

where each word w is included in the weighting vector of the document d and cluster c.

The cosine treats both vectors as unit vectors by normalizing them, giving a measure of the angle between the two vectors. It does provide an accurate measure of similarity but with no regard to magnitude (i.e. if a word occurs in both vectors - no matter the number of occurrences - they hint at the same direction). If magnitude (euclidean distance) was taken into account, the results would be quite different.

Based on cosine similarity, the decision of whether the input document belongs to an existing cluster is taken by the use of a pre-defined threshold parameter. If the similarity is below a certain threshold the document is considered to discuss a new topic and is mapped to a new cluster. Otherwise, the document is grouped to an existing cluster with the highest similarity score. Many improvements of that technique have been applied. Some of them can be found in the survey of clustering data mining techniques (Berkhin, 2002).

The threshold-based algorithm was invented by the TDT team at the Carnegie Mellon University (CMU) to decide whether a new document is another story of one of the detected events or it belongs to a new event of its own. The simplest form of the proposed algorithm, which relies only on pair-wise document similarity information, is portrayed in Algorithm 1.

Similarity functions can be used in conjunction with a wide variety of traditional clustering techniques; the most prominent among them are partitioning, hierarchical and density-based algorithms. These techniques will be discussed in the next subsections.

2.7.1 Partitioning Algorithms

The partitioning methods divide the initial dataset into sets of several clusters, where each object belongs to only one cluster. They create a one-level, un-nested partitioning of the data points where the number of clusters is given as input. Each cluster is described by a centroid or a representative, which is a summary of the objects. The precise form of the centroid depends on the type of the clustered objects. In case of real-valued data the arithmetic mean of each attribute is calculated. In case of documents the mean value is calculated based on the weighting metric.

There are a number of partitioning techniques, but we shall only describe some of them applied in the field of document clustering. The most representative algorithm of this category is k-means (Hartigan and Wong, 1979). The basic k-means algorithm follows the concept of centroid to represent the cluster. At the beginning, it selects K points as the initial centroids and it assigns all points to their closest centroid. Then, it recomputes the centroid of each cluster and repeats the same procedure until it converges and there are no changes to the centroids. Fuzzy c-means is another well-known unsupervised clustering technique that it also requires the number of clusters as input (Win and Mon, 2010). It differs from k-means as it calculates the document's degree of membership to every existing cluster. An experimental study on text documents showed that fuzzy clustering is a more stable method, as it produces better results on almost all datasets (Singh et al., 2011). Some other studies compare these algorithms with hierarchical approaches discussed in the next section (Steinbach, 2000).

2.7.2 Hierarchical Algorithms

The hierarchical methods produce a hierarchy of clusters, where each cluster is nested into another. The hierarchy, usually presented in a dendrogram, can be considered as bottom-up or top-down. Both techniques are met in literature and known as agglomerative and divisive methods respectively.

Agglomerative methods begin with n clusters for a dataset of n objects. Then, they choose and merge the closest two clusters into one. This process is repeated until only one cluster finally remains. On the other hand, divisive methods begin clustering using the opposite direction. They first put all n objects into one cluster and then they split the initial cluster into two smaller. In each step, a cluster is chosen and split up into two. This process continues until n clusters are produced. Those two methods are known for their weakness to revise a previously taken decision. Once a cluster is merged or split, it can never be separated or regrouped. That irrevocable decision is their major defect. Furthermore, the complexity of agglomerative clustering is, in the general case, O(3n), which makes them inappropriate for large datasets. Also, divisive methods have a worse complexity of O(2n) in the task of exhaustive search.

In the field of document clustering, many agglomerative methods have been proposed (Fung et al., 2003; Xu and Wang 2005). Among them, the single linkage method SLINK computes the similarity of two clusters as the similarity between the most similar pair of documents each of which belongs to different clusters (Sibson, 1973). The complete linkage method CLINK defines the similarity of a pair of clusters as the similarity between the most dissimilar documents, one in one cluster, and one in the other (Defays, 1977). The definition of clusters distance is much stricter than for single linkage. In the latter, two clusters may be forced together due to single documents being close to each other, even though the other elements might not be similar. This observation leads to chaining phenomenon where a small number of large clusters are created. On the other hand, complete linkage provides large number of small, tightly bound clustering.

A more extensive survey of hierarchical algorithms as well as a comparative analysis between well-known hierarchical algorithms can be found (Zhao and Karypis, 2002; Steinbach, 2000).

2.7.3 Density-Based Algorithms

A wide variety of density-based algorithms has been proposed in literature (Ester et al., 1996). These algorithms rely on a density-based notion of clusters and they inherently carry the notion of noise. This means that density of points within a cluster is considerably higher than outside of the cluster. Respectively, the density inside the area of noise is lower than inside of any cluster.

The key idea of all density-based algorithms is that every cluster has a minimum number of points, where the distance among them is below a specified threshold.

A big advantage of that technique is that it does not need an a-priori knowledge of the number of clusters and it is able to identify clusters of arbitrary shape and size. The exposure of non-linear shapes structures is based on the concept of density reachable and density connected data.

2.8 INFORMATION RETRIEVAL IN RECOMMENDER SYSTEMS

Some of the main responsibilities of information retrieval (IR) systems include data storage, organization, representation and easy access. In this research study, one form of IR used is the document retrieval based on an input query. Several algorithms have been implemented for the same reason. The algorithm accepts an input query and match it with documents stored and rank the output/results based on their similarity score with respect to the given query. These algorithms rely on matching the indexed documents, maintain the information concerning term frequencies and positions against the individual query terms. A score is assigned to each document based on its similarity value. A query term’s score with respect to a document is high for high frequency of appearance. IR involves different steps in preparing the documents for analyzing and calculating the similarity scores.

2.8.1 Text Preprocessing

The preprocessing procedure in text mining has a great impact on the quality results in information retrieval. It consists of three major steps in which plaint text is given as input and a vector of tokens is returned as output. These steps are listed and described below and has been applied to majority of text mining literature.

1) **Tokenization**: This step splits sentences into typical words called tokens. In some cases, more sophisticated techniques are applied, such as grammatical structure extraction.
2) **Stopwords Removal:** This step removes words that are commonly used and met in most text documents yet they do not have value for information retrieval purpose.
3) **Stemming:** This step converts the words to their root, base or stem form. Algorithms such as Porter's Stemmer and Lovin's stemmer have been studied in the literature of text preprocessing.

2.8.2 Information Extraction

The extraction of relevant information is very vital in information retrieval as the quality of recommendation is highly dependent on the appropriateness of the input features. Feature selection and feature extraction are used to reduce the dimensionality of the input data and provide better text representation.

- Feature Selection: This process locates the best minimum subset of the original features in order to reduce the initial dimensions.
- Feature Extraction: this process transforms the data of a high-dimensional space into a fewer-dimensional space.

2.8.3 Information Weighting

There are many information weighting techniques used in text mining. The most common weighting scheme used for terms within a text document is the Vector Space Model (VSM). VSM is an algebraic model for representing text documents as a vector of identifiers. The prevailing technique of the term weighting is Term Frequency – Inverse Document Frequency (TF-IDF). In a normal model, the TF component represent the number of times a term t appears in a specific document d and IDF is intended to account for common terms in the collection of documents. Below are the equations TF-IDF as used by Kumaran and Allan (2005):

$$w_{t,d} = tf * idf \tag{5}$$

$$tf = \log(t+1) \tag{6}$$

$$idf = \log\left(\frac{docCount + 1}{documentFreq + 0.5}\right) \tag{7}$$

where docCount is the number of documents in the corpus and documentFreq is the number of documents containing the term.

2.9 SIMILARITY MEASURES IN RECOMMENDER SYSTEMS

In information retrieval and data mining, similarity measure is described as an algorithm that determines the degree to which entities match (Oviatt and Cohen, 2000). The selection of similarity measure to use depends on the type of data and the objectives of the researcher. Some common measures used to determine similarity are described in the following subsections.

2.9.1 Euclidean Distance

It is the regular distance between two points and can be easily measured with a ruler in two - or three - dimensional space. Euclidean distance is widely used in clustering problems, including clustering text. It satisfies all the above four conditions and therefore is a true metric. It is also the default distance measure used with the K-means algorithm.

Measuring distance between text documents, given two documents d_a and d_b represented by their term vectors $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$ respectively, the Euclidean distance of two documents is defined as:

$$D_E(\overrightarrow{t_a}, \overrightarrow{t_b}) = (\sum_{t=1}^{m} |w_{t,a} - w_{t,b}|^2)^{1/2} \tag{8}$$

where the term set is $T = \{t_1, \ldots\ldots, t_m\}$. As mentioned previously, we use the $tfidf$ value as term weights, that is: $w_{t,d} = tfidf$

2.9.2 Cosine-based similarity

In cosine-based similarity, two items are thought of as two vectors in the m dimensional user-space. The similarity between them is measured by computing the cosine of the angle between these two vectors. Formally, in the $m * n$ ratings matrix, similarity between items i and j, is given by:

$$sim(i,j) = \cos(\bar{\imath}, \bar{\jmath}) = \frac{\bar{\imath} * \bar{\jmath}}{|\bar{\imath}| * |\bar{\jmath}|} \qquad (9)$$

where $\bar{\imath}$ and $\bar{\jmath}$ are m-dimensional vectors over the user-space. In documents similarity, m-dimensional will be term set.

Each dimension represents a term with its weight which ranges from 0 to 1. An important property of cosine similarity is its independence of document length.

2.9.3 Jaccard distance

It is described as the intersection of objects divided by the union of the same objects. The similarity index ranges from 0 to 1. It is 0 when the objects are completely different and it's 1 when the object are completely similar. The similarity is given by:

$$Jaccard(Q,D) = 1 - Sim(Q,D) = \frac{|Q \cap D|}{|Q \cup D|} \qquad (10)$$

where Q and D are the objects being compared for similarity. The similarity index is 1 when $Q = D$ and 0 when Q and D are disjoint.

2.9.4 Correlation-based similarity

In this case, similarity between two items i and j measured by computing the $Pearson\text{–}r$ correlation $corr_{i,j}$. To make the correlation computation accurate co-rated cases. (i.e., cases where the users rated both i and j) must be isolated, as depicted in equation 11. Let the set of users who both rated i and j be denoted by U,then the correlation similarity is given by:

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_{\imath})(R_{u,j} - \bar{R}_{\jmath})}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_{\imath})^2}\sqrt{(R_{u,j} - \bar{R}_{\jmath})^2}} \qquad (11)$$

where $R_{u,i}$ is the ranking of user u on item i, R_i denotes the average ranking of the i^{th} item and $R_{u,j}$ is the ranking of user u on item j, R_j denotes the average rating of the j^{th} item.

2.10 EVALUATION METRICS IN RECOMMENDER SYSTEMS

The evaluation of a recommender systems should be determined by the goals that the researcher wants to achieve. To analyse the performance and quality of a recommender system, it is essential to assess the recommendations utilizing evaluation metrics. Evaluation metrics assist to compare numerous arrangements recommended in the literature and subsequently, proposals have been improved bit by bit (Cacheda et al., 2011). The current evaluation metrics have a standard formulization that is utilized for the testing and assessments of the suggestions (Bobadilla et al., 2013). There are several criteria that lead to the selection of the evaluation metrics, but in this section, the focus will only be on the following: accuracy, quality and performance and stability.

2.10.1 Accuracy

Accuracy is a measure of how effective the recommender system's predictions are to real results. Researchers normally utilize the estimations of most normally utilized error forecast measurements like mean absolute error (MAE) to measure the accuracy of a recommender system. MAE is measured for items that the user rated and can be calculated as in equation 12 below (Jannach et al., 2013):

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \tag{12}$$

where p_i and q_i are the ratings-prediction pairs of user i and N is the total number of corresponding pairs.

Another measure that can be used to measure accuracy is the root mean square error (RMSE). It is a statistical metric slightly different from MAE. After calculation of the rating-prediction difference, the power of 2 is taken. After adding them up and dividing them by the total number of rating-prediction pairs and taking square root of it, RMSE is found. In Jannach et al. (2013), RMSE is given by equation 13:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (p_i - q_i)^2}{n}} \tag{13}$$

where p_i is the prediction of user i, q_i is the real rating of user i and n is the number of ratings-prediction pairs.

2.10.2 Quality and performance

User satisfaction does not only depend on the accuracy, but also on the quality and performance of a recommender system. The quality and performance of a recommender can be measured using metrics like Precision, Recall and F-measure. Precision shows the fraction of retrieved documents that are relevant. Recall measures the fraction of relevant documents retrieved from all relevant documents in the dataset. Equation 14 and 15 illustrate further on description of precision and recall respectively.

$$Precision = \frac{relevant\ documents \cap retrieved\ documents}{retrieved\ documents} \tag{14}$$

$$Recall = \frac{relevant\ documents\ \ \cap retrieved\ documents}{relavant\ documents} \tag{15}$$

As precision and recall have an inverse correlation, mostly a combination is used for optimization. Recall will score 100% if all relevant documents are retrieved plus infinitely many irrelevant ones. Precision will score 100% if the model managed to retrieve.

The F-measure (or F1-score) is defined as the harmonic mean between precision and recall. The harmonic mean is used because we are dealing with ratios and percentages, as it tends toward the least number, minimizing the impact of large outliers and maximizing the impact of small ones (Nadeau and Sekine, 2007). In other words, both Recall and Precision have to be of a high value to have a high F1-score, making harmonic mean a better fit than the arithmetic one. Equation 16 illustrate how F1-score is calculated.

$$F1 - score = 2 * \frac{Precision\ * Recall}{Precision\ +\ Recall} \tag{16}$$

2.10.3 Stability

A user has a more trust in a RS when the recommendations generated by the system best match the user's preferences. A RS is known as stable when the recommendations generated do not deviate over a short period (Adomavicius and Zhang, 2012). The metric defined for the evaluation of the stability of RS is Mean Absolute Shift (MAS) (Verbert et al., 2012). The MAS metric consists of a set of known ratings R1 known by the user and a set of unknown ratings Q_1. After a period of time, the user rated some of the unknown ratings and the new recommendations Q_2 are generated by the system. Now, MAS can be calculated using equation 17 (Verbert et al., 2012).

$$MAS = \frac{1}{|Q_2|} \sum_{(u,i)\in Q_2} |Q_2(u,1) - \ Q_1(u,1)| \tag{17}$$

2.11 CHAPTER SUMMARY

In this chapter, the focus was on reviewing literature in recommender systems. First, the literature on the background of recommender system was done for better understanding of these systems. Different filtering algorithms such as content-based filtering and collaborative filtering algorithms were discussed. This research study uses content-based filtering algorithm to make recommendations. Furthermore, some characteristics of mobile recommender systems were discussed. Also, the challenges that face mobile recommender systems we studied. Then location awareness recommender systems and service provided by them were discussed.

In addition, existing job employment recommender system were also studied to gain more knowledge on these systems as that is the major part of this research study. Information retrieval in recommender systems is also studied in this chapter. It included text preprocessing, information extraction and information weighting. Then, the similarities in recommender systems such as cosine similarity and Jaccard distance are discussed. Cosine similarity is the preferred similarity measure in the study. Lastly, the evaluation metrics in recommender systems are reviewed. The following chapter will focus on the research methodology used throughout this research study.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 INTRODUCTION

This chapter presents the research methodology and algorithms that are applied in this study. Research methodology can be defined as logical, academic analysis of processes used in a specific research area. Its major classifications involve quantitative and qualitative research methods (Ishak and Alias, 2005). In this research, qualitative research methodology is applied through literature investigations and the implementation and evaluation of the recommender system prototype. A client-server architecture prototype of the recommender system was implemented for the base of evaluation of this research study. Therefore, the recommender system was evaluated on the client and server side.

From the literature analysis, there are three main filtering algorithms in the field of recommender systems. These include collaborative filtering, content-based filtering (CBF) and hybrid. In this research the focus is solely on content-based filtering. CBF recommends objects which have the same content as the ones which the target user prefers. For the sake of our job employment recommender system, CBF algorithm is used to recommend the most similar resume to the job description at the same time finding the proximity of the job seeker to the area where the job will be done.

3.2 SYSTEM ARCHITECTURE

The prototype of a Mobile Proximity Job Employment Recommender System (MPJERS) developed uses client-Server architecture as demonstrated in Fig 3.1. The client side comprises of two components: (i) A mobile application and (ii) the location manager. A mobile application requests a temporary job seeker to be registered in the application. A personalized profile for each registered job seeker is generated and the login details are required each time the job seeker wants to access the application. All registered job seekers are also required to upload their resumes on the mobile application which will be stored in the database under the server side and used to recommend a suitable job to a suitable candidate. A location manager is used to pinpoint the temporary job seeker’s current proximity to the place where the job will be done.

On the server side, the essential part is the recommendation engine consisting of offline and online subsystems. The offline subsystem maintains a USER_DATA database, which stores the records of all job seekers that are registered in the system. The user profiles for all job seekers are then generated and stored in a different database (USER PROFILE). The online subsystem on the other hand, maintains the USER PROFILE database and JOB DATA database containing the temporary job's information, such as employer name, employment location, temporary job position and, skill requirement. When receiving a job request, the online subsystem computes the similarity between the user profile and the job data for a specified temporary job and then finds the instantaneous proximal location of the temporary job seeker to the employment location through the location manager and generates a list of possible recommendations.

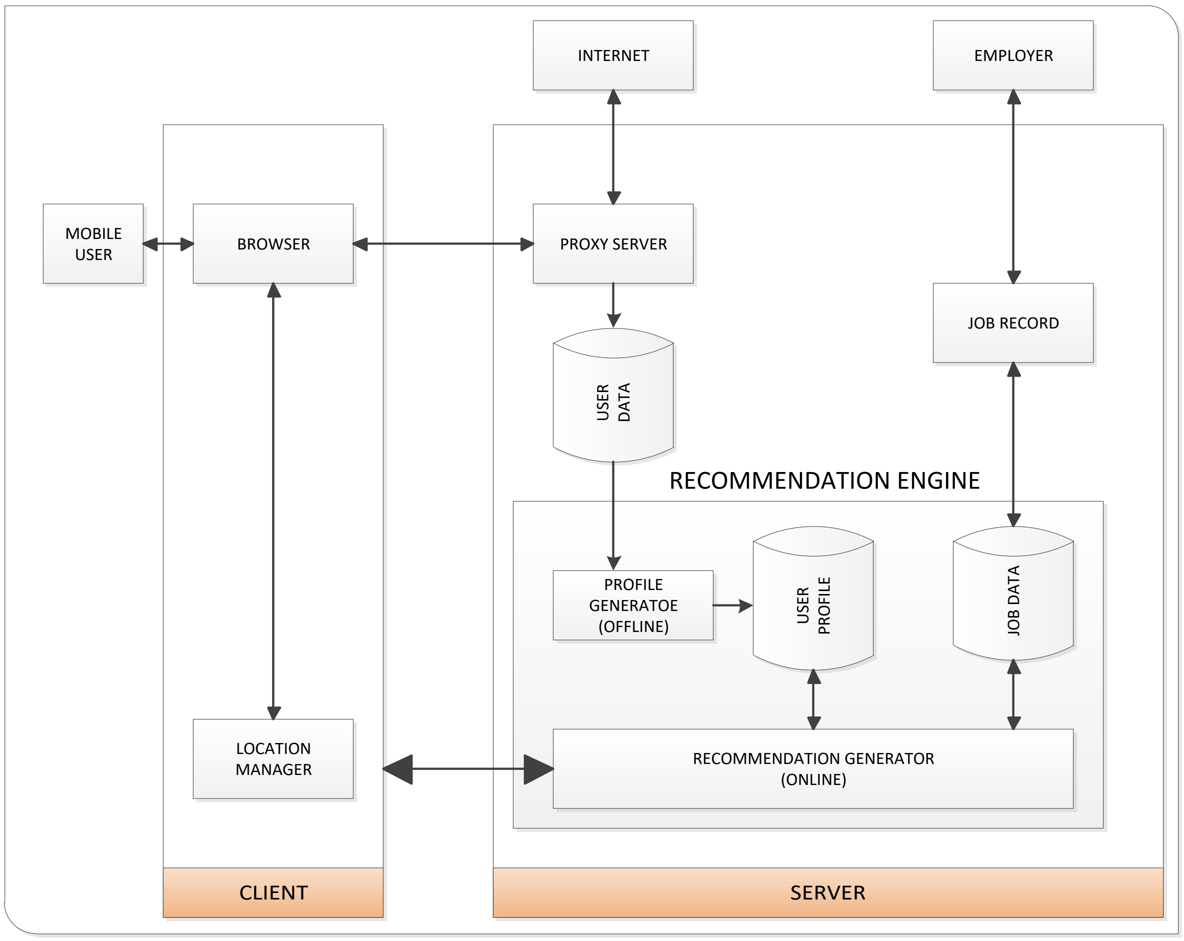


Fig 3.1 - Mobile Proximity Job Employment Recommender System Design Architecture (Zuva, T and Zuva, K, 2014; Yang et al, 2008)

3.3 CONTENT-BASED FILTERING

Since the purpose of this study is to enable the initial selection of suitable temporary job seekers to a temporary job at a particular place and vice versa, this study uses CBF to recommend the most suitable resume to the job description. The principle of a content-based recommendation suggests objects which have the same content as that of the target user (Liu et al., 2015). When recommending people to jobs, CBF make recommendations based on the content of resumes and job descriptions text documents. Thus, the content of a resume document in the user profile was parsed.

The process of CBF is selecting the same features between resumes and job description documents as well as computing their similarity to compare them. The predicted outcome will include a list of applicants and the recommended jobs arranged by their similarity values. This implies that the key processes in CBF includes selecting relevant features and computing similarity. Throughout the selection of features, it is essential to find similar features. Also, it is vital to consider the effect of recommendation in accordance with the scientific investigations in the job recruitment domain or target users' preferences. Features that are selected are then presented in a vector format so that calculations of similarity can take place. Selected features in this study are represented in a vector space model (VSM) format and similarity was calculated through cosine similarity.

3.4 FEATURE SELECTION

In this study, the feature selection was done in two major categories in order to gather all data that may possibly be used to match the entities (Job profile and Jobseeker profile) used in the MPJERS. The first category was the review of the job features that was carried out to understand what information is passed to the system by the employer as the basis for recommendation. The second phase was an inspection of the candidate features that was then carried out to understand the interaction of the job seeker with the system and what information the job seeker passed to the system.

The features relating to the job used in MPJERS are: qualification, required skills, experience, employer/company name, industry field, temporary position offered, payment rate, and employment location. Collectively, these selected features define the temporary job for the proposed MPJERS. Fig. 3.2 illustrates the relevant features for the temporary job used in the MPJERS.

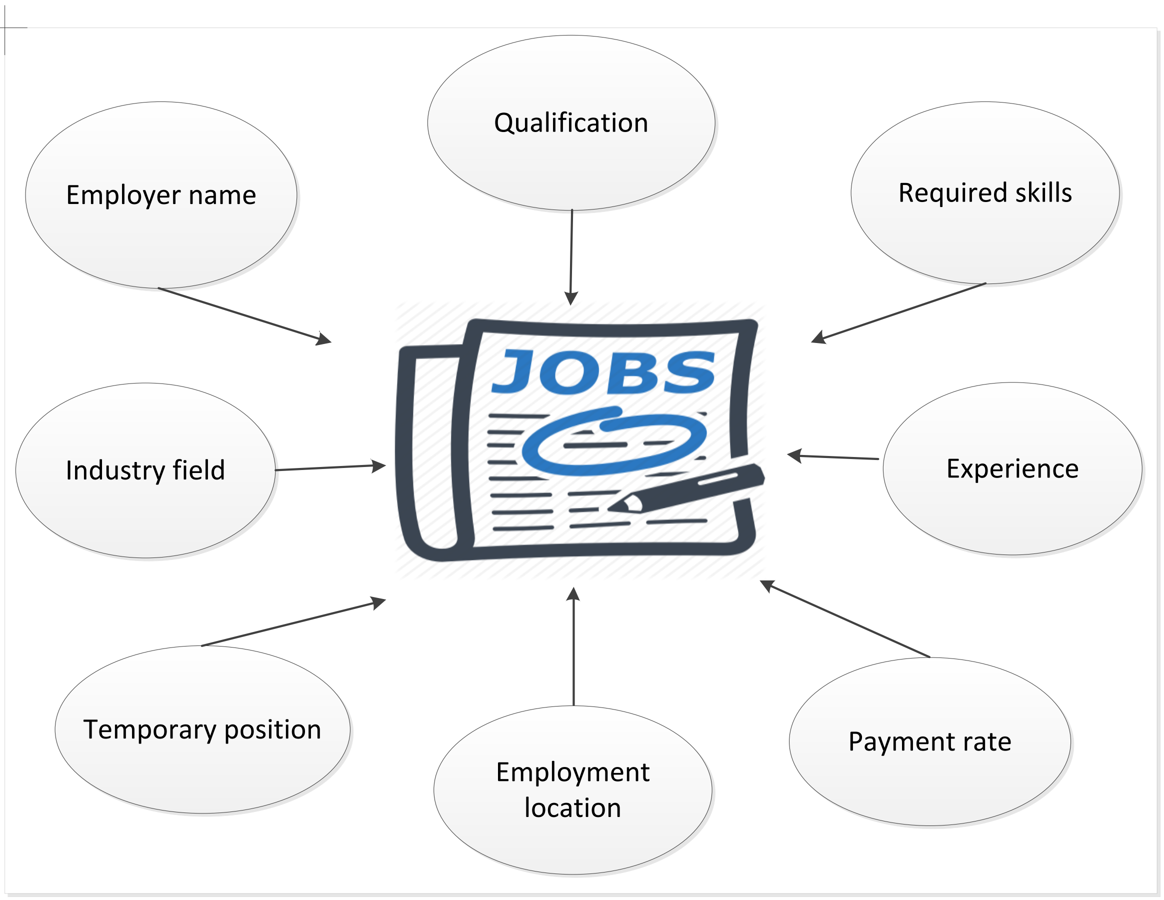


Fig 3.2 - Features relating to the job

For the job seeker, the features that were considered for recommendation are: age, gender, educational background, experience, skill, current location and current employment. These selected features collectively define the temporary job seeker for the proposed MPJERS and are illustrated in Fig 3.3.

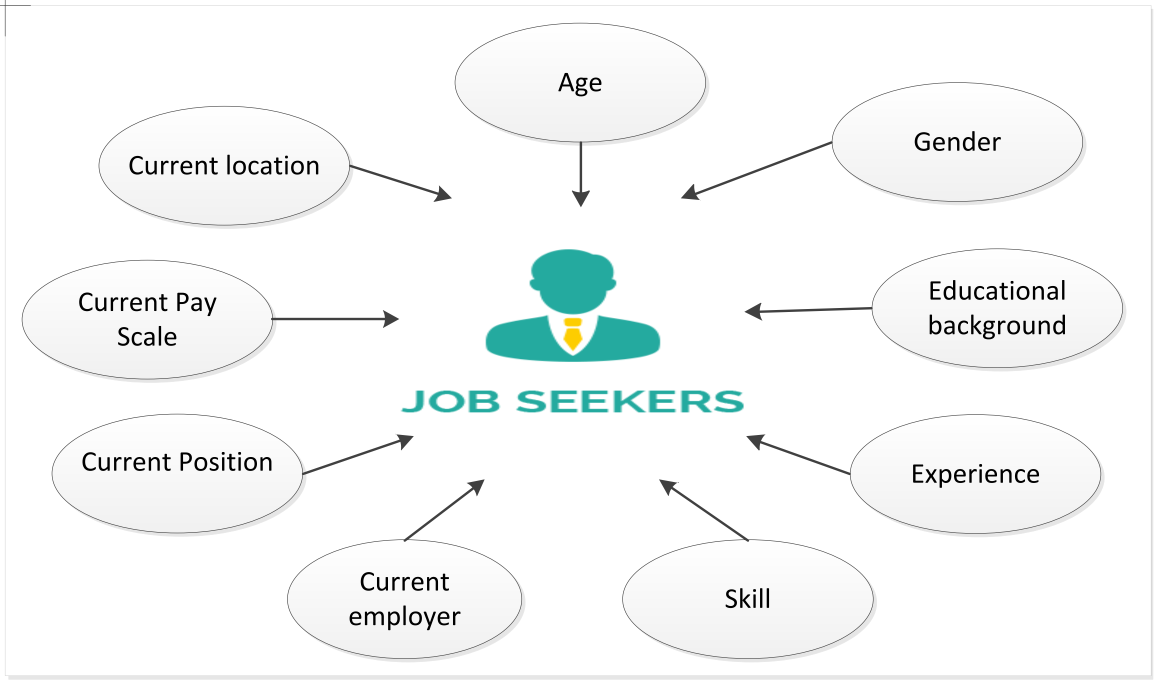


Fig 3.3 - Features relating to the job seeker

3.5 VECTOR SPACE MODEL

According to Montaner et al. (2003), the vector space model (VSM) is a standard algebraic model commonly used in information retrieval (IR). It deals with a text document as a collection of words, without considering word order and grammar. It represents both documents and queries by term sets and measure the similarities between documents and queries. In this research study, VSM uses Term Frequency – Inverse Document Frequency (tf-idf) to weight the terms. Each document is represented as a vector of tf-idf weights. Queries are also considered as documents. Cosine similarity is used to compute similarity between document vectors and the query vector.

3.5.1 Text Document Preprocessing

Preprocessing ensures that text documents must be structured in a suitable form for information retrieval and the data sets that are used must lead to the best performance and excellence for the models obtained by information retrieval processes. Text document preprocessing involves three major steps described below:

3.5.1.1 Tokenization

The main objectives of tokenization are to split texts into plain processing units, interpret and collect isolated tokens to produce higher level tokens. The data must be processed in the three operations: the first operation is to convert documents to word count which is equal to bag of word (BOW). The second is removing empty sequence, i.e. this step comprises cleansing and filtering (e.g., whitespaces, collapsing, stripping extraneous control characters). Finally, each input text document is segmented into a list of features which are also called tokens, words, terms.

3.5.1.2 StopWord elimination

The main objective of stopWord elimination is to remove words that occur in most of the documents. Those words include articles (a, an, the, etc), conjunctions (and, or, but, etc), prepositions (in, on, of, etc), pronouns (I, you, it, etc) and possibly some nouns, verbs, adverbs and adjectives (make, thing, etc). Such words are referred to as stopWords and they do not have any value for retrieval purpose.

3.5.1.3 Stemming

The main objective of stemming is to replace all variants of the word with the single stem of the word. Variants include past tense suffixes, gerund forms (ing – form), plural, third person suffixes, etc. This process removes affixes (prefixes and suffixes) from a word and retains the word to its original form call stem.

3.5.2 Term Document Matrix

A term document matrix is an approach of demonstrating documents in a matrix format in which each column represents term vectors across all the documents and rows represent documents across all the terms. The cell values represent the number of occurrences of each term in the corresponding document.

3.5.3 Term Weight

Once the term document matrix is generated, the term weight for all the matrices in all the CVs are calculated. It is also vital to compute the term weighting because terms which uniquely describe the CV need to be well known. It should be noted that words which appear in most CVs might not represent the relevant CVs whereas words that appear less frequent might define a relevant CV. To find most relevant CVs, a technique called term frequency – inverse document frequency (TF-IDF) is used. It gives lower weights to terms which occur frequently across all the CVs and gives higher weight to the terms which rarely occur in all other CVs. Term frequency (TF) is simply the frequency of a word in a document. Inverse document frequency (IDF) is the inverse of the document frequency among the whole corpus of documents. TF-IDF negates the influence of high frequency words in determining the significance of a document.

To measure how important a keyword may be or the number of occurrences a keyword appears in a document, log is used to dampen the effect of high frequency words. Jain et al. (2017) demonstrated how the weighted term frequency can be measure using equation 18.

$$w_{td} = \begin{Bmatrix} 1 + log_{10} tf_{t,d} & \text{if } tf_{t,d} > 0 \\ 0\,, & \text{otherwise} \end{Bmatrix} \tag{18}$$

where w_{td} is the weighted term frequency, and tf_{td} is the term frequency.

Then the IDF is computed by taking the logarithmic inverse of the number of occurrences of a keyword among the whole collection of the documents. So, from the total of all the CVs returned by the search query, the IDF will be calculated as show in equation 19 (Jain et al., 2017).

$$IDF = log_{10} \frac{N}{DF} \tag{19}$$

where N is the total number of documents and DF is the total number of the keyword in all the documents.

Together the weight of a term is the product of the two weights; the TF weight and the IDF weight as illustrated in equation 20 (Jain et al., 2017).

$$tfidf = (1 + log_{10} tf_{td}) * \left(log_{10} \frac{N}{DF}\right) \tag{20}$$

3.6 DOCUMENT CLUSTERING USING K-MEANS ALGORITHM

Grouping documents into several collections in a manner that similar documents be located in a matching set and other dissimilar documents in different sets is the main objective of document clustering. For text document clustering, term frequency is used to cluster documents which have similar words in a collection. Words are used to gauge contents in documents. Normally, text documents are characterized by word vectors. Each document vector comprises of all terms which occur in the whole document corpus. Term frequency (tf) or term frequency

– inverse document frequency (tf-idf) are used as measures for each entry in a term vector. To calculate similarity, documents characterized by term vectors are compared to one another through the use of similarity measures like Pearson similarity, cosine similarity or adjusted cosine similarity. Documents are allocated to specific clusters based on their degree of similarities.

The most well-known hierarchical algorithms are single-link and complete-link; the most popular and the simplest partitional algorithm is K-means. Since partitional calculations are favored in design acknowledgment because of the idea of accessible information, our scope here is centered on these calculations. K-means has a rich and various history as it was autonomously found in numerous research papers and commercial projects. Even though K-means was first proposed more than 50 years back, it is as yet a standout amongst the most broadly utilized calculations for grouping. Simplicity of execution, effortlessness, proficiency, and observational achievement are the primary explanations behind its fame.

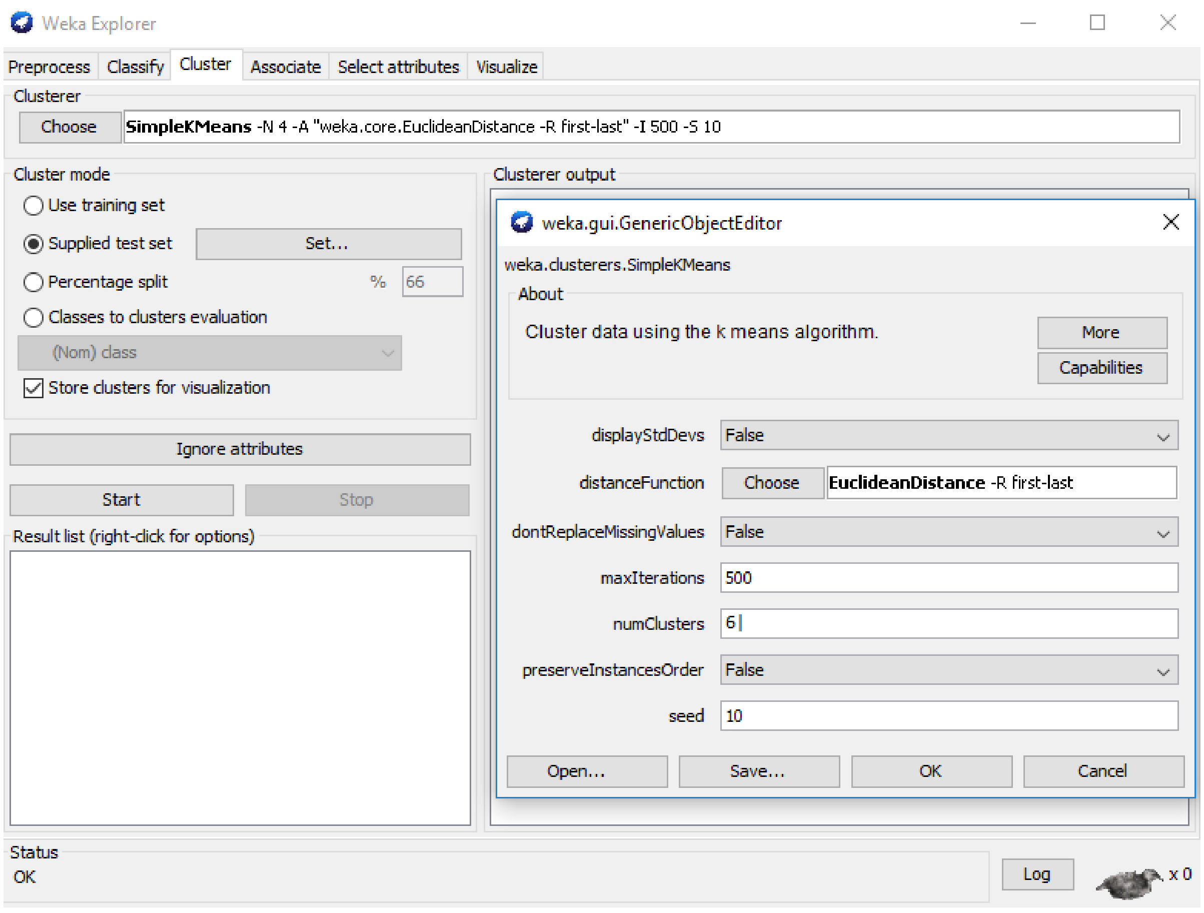
In data mining, k-means clustering is described as a technique of cluster analysis which separates n observations into k groups (Gao et al., 2015). Every observation fits into a group that has the closest mean. As one of the most straightforward, unsupervised machine learning algorithms, k-means take care of notable grouping issues (Cacheda et al., 2011). The procedure follows an easy and straightforward tactic of arranging a collection of information into a certain number of clusters (k clusters for instance). It is thoughtful to represent k centroids, for each and every single group. Because different clusters give diverse results, the centroids must be kept in a clever way. In this way, it will be the best choice for the centroids to be kept far away from one another. The subsequent step is taking each observation that have space in a collection of information and group it to the nearest centroid.

At the point when no point is pending, the initial step and an early gathering age are finished. Now we must recalculate k new centroids as ban focuses of the groups coming about because of the past step. After we have these new k centroids, another coupling must be done between similar informational collection focuses and the closest new centroid. A loop has been generated. As a result of the loop, we may see that the k centroids change their position well-ordered until no more changes are done. As such, centroids don't move any more.

The algorithm is composed of the following steps:

1. Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object to the group that has the closest centroid.
3. When all objects have been assigned, recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

In this research study, simple K-mean algorithm was used to group similar resumes text documents together. From the dataset of resume text documents, six (6) clusters were generated using Euclidean Distance function from Weka as shown in Fig 3.4.



Clusterer

SimpleKMeans -N 4 -A "weka.core.EuclideanDistance -R first-last" -I 500 -S 10

Cluster mode

Use training set
[✓ Supplied test set]
Percentage split % 66
Classes to clusters evaluation
(Nom) class
[✓ Store clusters for visualization]

Result list (right-click for options)

Clusterer output

weka.gui.GenericObjectEditor

weka.clusterers.SimpleKMeans

About

Cluster data using the k means algorithm.

displayStdDevs	False
distanceFunction	EuclideanDistance -R first-last
dontReplaceMissingValues	False
maxIterations	500
numClusters	6
preserveInstancesOrder	False
seed	10

Status
OK

Fig 3.4- Resume clustering using K-means algorithm in Weka

The Euclidean Distance formula can be represented as in equation 21:

$$dist = \sqrt{\sum_{k=1}^{n} (P_k - q_k)^2} \qquad (21)$$

where n is the number of dimensions (attributes) p_k and q_k are, respectively, the k^{th} attributes (components) of records p and q.

3.7 COSINE SIMILARITY

The distance between two vectors is computed by finding the cosine angle between the two vectors. In the same way, cosine angle between each resume vector and the query vector (Job description) is computed to find how close a CV is to a query. To find a relevant resume to the query term, similarity score between each resume vector and the query term vector is calculated using cosine similarity as illustrated in the equation 22 as derived from Liu et al. (2016). The resume with the highest similarity scores will be considered as relevant documents to the query.

$$Cos\left(U_i\, U_J\right) = \frac{U_i . U_j}{|U_i| * |U_j|} = \frac{\sum_1^n (tfidf_{Ui} * tfidf_{Uj})}{\sqrt{\sum_1^n (tfidf_{Ui}^2) + \sum_1^n (tfidf_{Uj}^2)}} \quad (22)$$

where U_i represent the query vector and U_j represent the CV vector.

After plotting the term document matrix, each resume vector represents a point in the vector space. In Fig 3.5, CV1 and CV2 represents 3 points in the vector space. Each CV is compared with the query by calculating the cosine angle between them. From Fig 3.5, it is obvious that the angle between CV2 and the query is smaller than the angle between CV1 and the query. This implies that CV2 is closer to the query than CV1, making CV2 more relevant to the query than CV1.

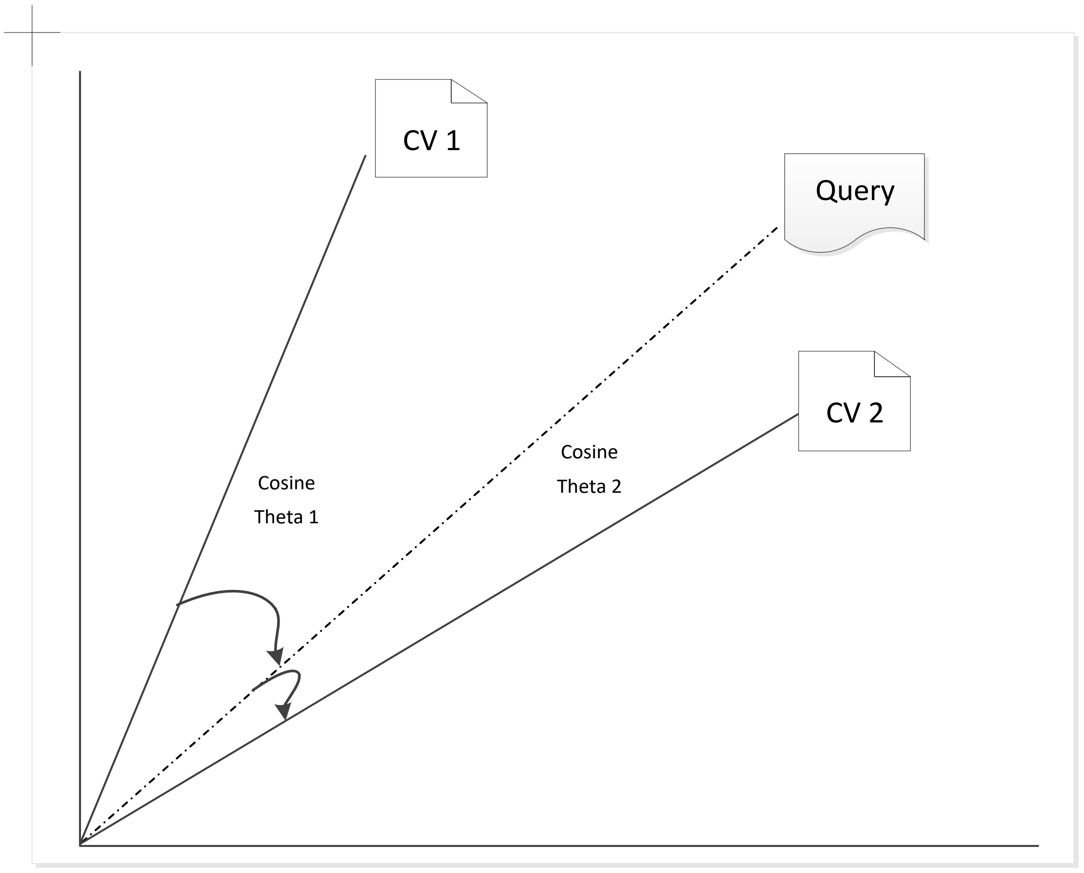


Fig 3.5 - Cosine similarity

3.8 GLOBAL POSITION SYSTEM

After finding the most suitable candidates for the temporary job through the use of vector space model and cosine similarity, the recommendation generator on the server side generates a recommendation list and then links up with the location managers on the client side to find the current positioning of the mobile user to the area where the job will be done. Google maps coordinates are used to calculate the distance between the candidates on the recommendation list and the place where the job will be done. In this research, the distance between the candidate and the area where the job will be done is calculated using the equation 23:

$$d_{ul} = 2 * a\sin(sqrt((lat_1 - lat_2))^2 + \cos(lat_1) * \cos(lat_2) * (\sin((lon_1 - lon_2)/2))^2 \qquad (23)$$

where (lat_1, lon_1) and (lat_2, lon_2) are the coordinates for the mobile job seeker current location and the area where the job will be done respectively.

It is of vital importance to note that lat and lon represent latitude and longitude respectively. North latitudes and west longitudes are considered as positive and south latitudes and east longitudes are taken as negative.

3.9 SYSTEM PROTOTYPE EVALUATION

There are several evaluation methods that can be applied and used in recommender systems. These methods differ depending on the objective that the researcher wants to achieve. The evaluation of this study is done offline. The datasets that were found to evaluate the server side of the MPJERS. It is imperative to note that the recommender system in this study can be seen as a client server architecture whereby the client side is the mobile job seeker application and the server side is equipped with the database and recommendation system.

To effectively evaluate the performance of the MPJERS, precision and recall were calculated as measure of performance. In this research, precision is defined as the total number of all relevant documents divided by the total number of all retrieved documents and recall is defined as the ratio of correctly classified positive documents divided by the total number of positive documents. To quickly calculate precision and recall given the predicted labels from a model, the confusion matrix is used. For binary classification, a confusion matrix shows four different results namely:

I. True positive (TP): outcome is positive and it is predicted to be positive.
II. True negative (TN): outcome is positive, while it is predicted negative.
III. False positive (FP): outcome is negative and it is predicted to be negative.
IV. False negative (FN): outcome is negative while it is predicted to be positive.

Fig 3.6 shows the pictorial representation of the confusion matrix.

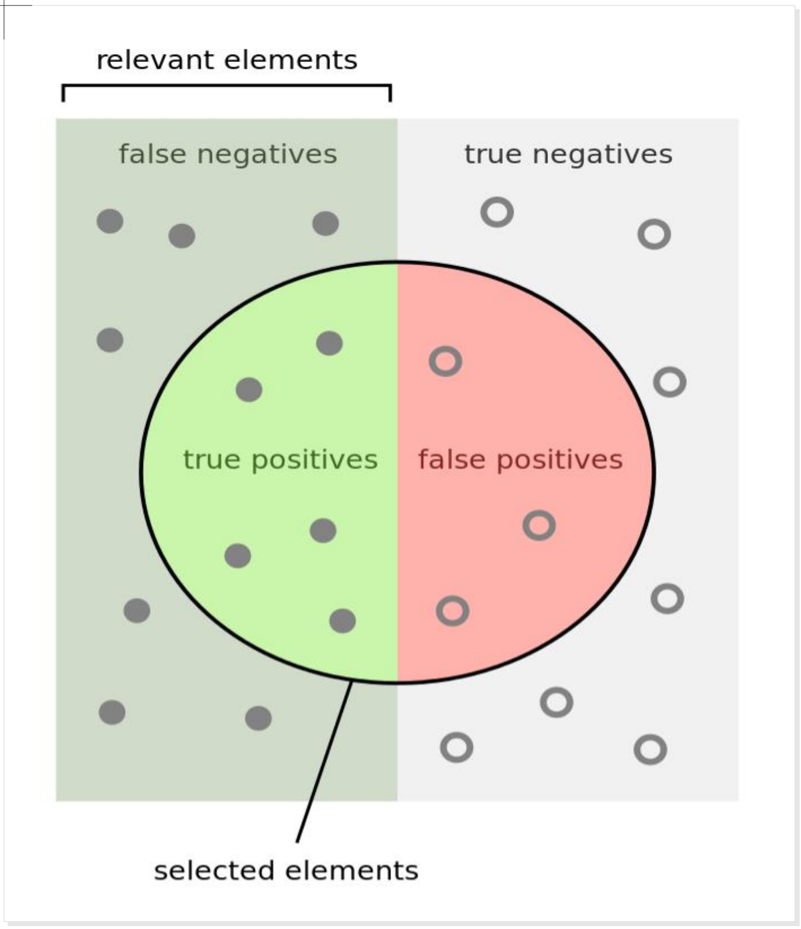


Fig 3.6 - Pictorial representation of the confusion matrix

A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. It allows the visualization of performance of an algorithm. Table 3.1 illustrates the confusion matrix.

Table 3.1 – Confusion Matrix

		Reality	
		Actually Good	Actually Bad
	Rated Good	True Positive	False Positive

prediction	Rated Bad	False Negative	True Negative

From the confusion matrix in Table 3.1, precision and recall can be calculated using the values in the matrix by applying equation 24 and 25 respectively.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{24}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{25}$$

It is of great importance to have a measurement that can represent both precision and recall because it is not easy to compare the two models with low precision and high recall or vice versa. Thus, to make both precision and recall comparable, the F-measure (or F1-Score) is used. F1- score helps to measure precision and recall at the same. The F1-score is defined as the harmonic mean between precision and recall and is calculated as in equation 26.

$$F1 - score = \frac{2 * Precision\ * Recall}{Precision\ +\ Recall} \tag{26}$$

To the user the scalar value of recall indicates the ability of the system to find relevant items as per query from the collection of different items. Precision demonstrates the capacity to yield pertinent items according to the query. The Precision-Recall graph gives a visual performance of the recommender system and the graph is demonstrated in Fig 5.6. These performance measures were used to evaluate the MPJERS.

3.10 CHAPTER SUMMARY

This chapter presented the research methodology used in this study. Firstly, a client – server architecture that was used to implement the prototype of MPJERS was described. The client side uses a mobile application which allows mobile job seekers to register their profile and log on to the system through a web server on the server side. The server side then stores the profiles of the mobile job seekers into the database and makes recommendations for a suitable job. Secondly, the proposed contend based filtering algorithm is explained and applied to the server side of the MPJERS prototype to make recommendations. The features of both the job seeker and the recruiter used for making recommendations were selected. The vector space model algorithm that uses TF-IDF was demonstrated. It also included the preprocessing of the text documents. Then, the resumes text documents were clustered into various clusters using the K-means algorithm. Cosine similarity was later used to compute the similarity between the resumes and job description text documents.

Furthermore, the GPS was used to pin point the current location of the suitable candidates and final recommendation was made based on how suitable candidates are closest to the area where the job will be done. Lastly, the proposed evaluation metrics is discussed. The next chapter will focus on the implementation and results of the system prototype used for the basis of evaluation the recommender system.

CHAPTER 4: IMPLEMENTATION AND RESULTS

4.1 INTRODUCTION

This chapter describes how the client-server prototype of a MPJERS was designed and implemented. A prototype is not complete in functionality or design, but serves as a proof-of-concept to be used as a basis for evaluation, conclusion and discussion in this research.

4.2 REQUIREMENTS

Based on the objectives listed in chapter 1, the findings during the initial research study and the analysis of existing systems, a set of requirements were formulated. In this section, requirements are divided into two categories namely, functional and non-functional requirements. Functional requirements are the descriptions of activities and services a system must accomplish whereas non-functional requirements are the descriptions of other features, characteristics and constraints that define an acceptable system.

4.2.1 Functional Requirements include:

- The client side should allow the user to register their profile and log in to the system.
- The client side should provide a user interface that allows the user to apply for a temporary job and sent the application to the server side.
- The client side should allow the user to modify and update their profiles.
- The client side should allow the user to send their current location.
- The server side should receive and store job applications from the client side.
- The server side should provide recommendations using content-based filtering.
- The server side should be able to depict the current location of the recommended job seeker.

4.2.2 Non-Functional Requirements include:

- The server side should provide accurate recommendations that match the best suitable candidate for a temporary job.
- The client side should be user friendly to capture user attention.

4.3 PROGRAMMING ENVIRONMENT

This section introduces the programming platforms used to build the client-server prototype of the MPJERS. To develop the client side, Java eclipse Integrated Development Environment (IDE) and Android System Development Kit (SDK) are used for development environment and user interface design respectively. Since the client side is developed using Java, the recommendation engine on the server side is also carried out using netBeans IDE for Java Enterprise Edition (Java EE).

4.3.1 Android Programming Environment

To develop the client side of the system prototype, the Android platform was utilized.

Android is an open source operating system for developing mobile devices, based on a Linux kernel (Chang et al., 2010). Applications are built in Java and run in their own process with their own instance of a Virtual Machine. Android was first implemented by Google but it is now part of the Open Handset Alliance. The android platform is classified into the following four sections:

1. Android Application: It is the first layer in the android platform stack and is made up of applications that are built-in the device or other third-party applications installed on the device.
2. Application Framework: This is the second layer in the android platform stock and it is built using java and provides high level services and APIs that are leveraged by the applications.
3. Libraries layer: The third layer is responsible for providing support to the core features.
4. Linux Kernel: The last layer provides support for security management, device management, memory management, network stack and process management.

Figure 4.1 shows the android platform architecture.

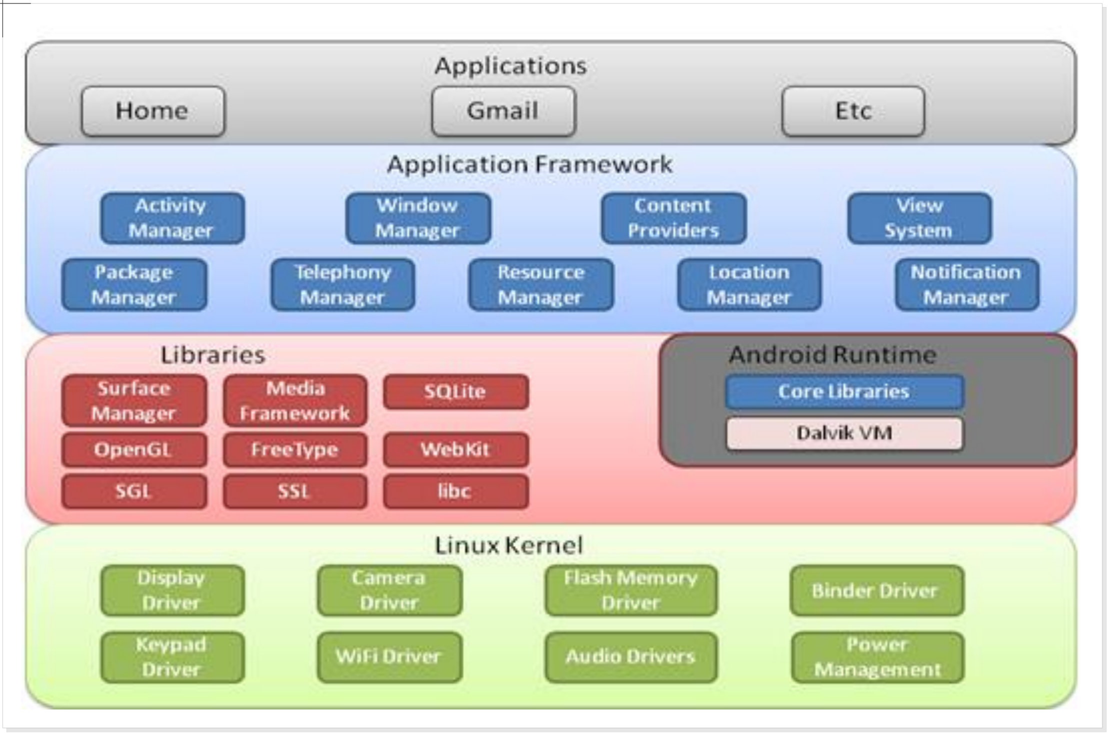


Fig 4.1 - Android Platform Architecture (Chang et al., 2010)

4.3.2 NetBeans IDE for Java EE

The server side of the MPJERS prototype was developed using the netBeans IDE for Java EE for a reliable and easy interchanging of data between the client side and the server side. Java EE applications are run on reference runtimes that can be microservices to handle transactions, security, scalability, concurrency and management of components it is deploying. Java EE is defined by its specification. The specification defines application programming interfaces (API) and their interactions. As with other Java community process specifications, providers must meet certain conformance requirements in order to declare their products as java EE compliant.

4.4 GLOBAL POSITIONING SYSTEM (GPS)

The GPS comprise of 24 satellites installed in space roughly 19300 kilometers (KM) above the earth surface (Warner and Johnston, 2003). These satellites revolve the earth once every 12 hours at a very fast pace of about 11200 km per hour. The satellites are equally spread out so that four satellites are reachable through direct line of sight from anyplace on the earth.

Each GPS satellite transmits a message which contains the satellite's present location and exact time. A GPS receiver combines the messages from different satellites to calculate the precise current location through a process called triangulation (Boneh et al., 2015). To determine a receiver's current position, only three satellites are needed. The fourth satellite is connected just to make sure a GPS is more accurate.

For a GPS device to function well, a connection to the needed number of satellites must be established initially. This procedure can take several seconds to several minutes depending on how powerful the receiver is (Boneh et al., 2015). For example, a smartphone's receiver will normally establish a slower GPS connection than a car's GPS unit. Most GPS devices use a certain kind of site caching to speed up GPS detection. A GPS device can decide quicker which satellites will be accessible the next time it scans for a GPS signal by remembering its previous location.

GPS technology is not ideal for indoor use because GPS receivers need a free pat to space (Chang et al., 2015). Thus, tablets, smartphones and other mobile devices frequently use other alternative means such as public Wi-Fi signals and cell towers, to determine location. This technology is used to supplement GPS technology when satellite connections is not accessible and it is called local position system (LPS)

4.5 CONTENT – BASED RECOMMENDER ENGINE

In this recommender engine, recommendations are centered on description of an item and a user's profile ideal choices. Keywords are used to describe the items. Besides, a profile of a user is built to state the type of item the user likes. This means the algorithm tries to recommend the items that are similar to the ones the user has liked before. For example, the system can recommend the same job as the user has applied for in the past. This tactic is fundamental in information filtering and information retrieval.

The main issue with content-based recommender engines whether they are capable to learn user first choice from users' activities about single content source and repeat them through other different content types. When the engine is restricted to recommending the content of the similar type as the user is already using, the value from the recommendation engine is considerably less when other services can be recommended.

4.6 DATASETS

The experiment conducted to evaluate performance of the recommender system was done offline. In order to conduct these experiments, the research required the datasets for job description text documents and job-related resumes\CVs for jobseekers to be available. Also, the dataset of longitude and latitude coordinates were required to evaluate the proximity. All job description text documents and job-related resumes were obtained from Kaggle website. Kaggle is an online community of data scientists and machine learners. Kaggle allows researchers to find data they need to do their data science work. About 190,000 public datasets and 200,000 public notebooks are freely available to researchers.

Datasets that were found consist of 203 job description text documents and 1431 resumes. Also, a CSV file consisting 8300 coordinates was found.

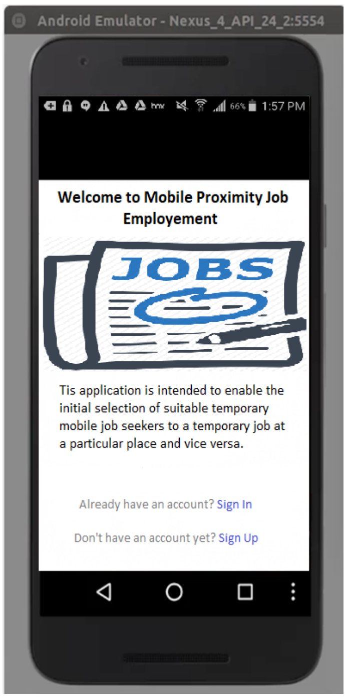
4.7 SYSTEM PROTOTYPE

To develop the system prototype of a MJERS, android software development kit (SDK) and eclipse IDE for Java EE are used. MySql and SqLite databases are used to store all the information that is needed for the functionality of this prototype. Xampp server is used at a server side.

4.7.1 Client Side

The client side is the link between the mobile jobseekers and the server. Its task is to gather information from the mobile jobseekers and to allow them to apply for a temporary job. The information is initially stored in the SqLite database and later sent to the server side, where user profile for each jobseeker is generated and stored in the MySql database together with the uploaded resume and later used for recommending the most suitable mobile jobseeker to a specific temporary job.

Fig 4.2 shows the startup screenshots for MPJERS client graphical user interface (GUI). Fig 4.2(a) displays the home screen with a brief description of the application and the links to either to sign in (login to an existing account) or sign up (create a new account). When the jobseeker clicks the sign in, the mobile job seeker will be required to enter the login credentials if they have already created an account before as shown in Fig 4.2(c). When the job seeker clicks sign up link, the screen in Fig 4.2(b) is displayed and a mobile jobseeker will fill all necessary details to create an account.


Android Emulator - Nexus_4_API_24_2:5554

1:57 PM

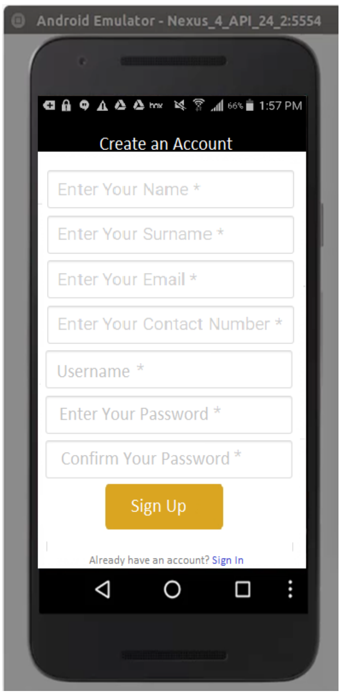
Welcome to Mobile Proximity Job Employement

Tis application is intended to enable the initial selection of suitable temporary mobile job seekers to a temporary job at a particular place and vice versa.

Already have an account? Sign In

Don't have an account yet? Sign Up

Fig 4.2(a) – Client Home Screenshot


Android Emulator - Nexus_4_API_24_2:5554

1:57 PM

Create an Account

Enter Your Name *

Enter Your Surname *

Enter Your Email *

Enter Your Contact Number *

Username *

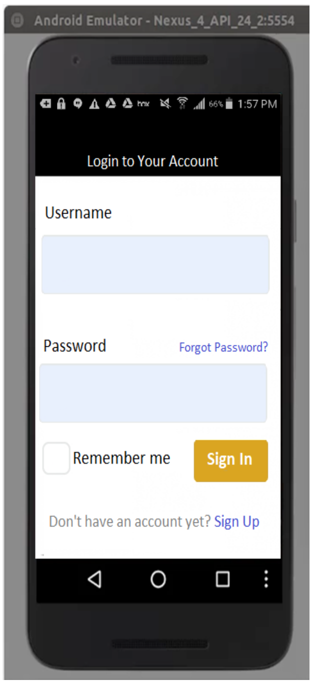
Enter Your Password *

Confirm Your Password *

Sign Up

Already have an account? Sign In

Fig 4.2(b) – Client Sign Up Screenshot


Android Emulator - Nexus_4_API_24_2:5554

1:57 PM

Login to Your Account

Username

Password

Forgot Password?

Remember me

Sign In

Don't have an account yet? Sign Up

Fig 4.2(c) - Client sign In Screenshot

Fig 4.2 - MPJERS Client Start-up Screenshots

After login to the application, a client can view more of the functionalities provided by the MPJERS. Fig 4.3 shows the functionalities that are provided by the MPJERS. Fig 4.3(a) shows a list of client services which include the following:

- Upload CV – A client can upload the CV through the application and then later used to recommend for a specified temporary job.
- Send Location – A client can send the current location through google maps so that the MPJERS can know the proximity of the job seeker to the place where the job will be done. Fig 4.3(b) shows how the current location is sent through google maps.
- Available Jobs – A client can view all available temporary jobs and the locations where the jobs will be done. Fig 4.3(c) shows how available jobs can be viewed.
- Recommend Jobs – A client can view specific jobs recommended for their profiles as per their uploaded CV and proximity to the place where the job will be done.
- Update Profile – A client can update their own profile in case the need to change anything like their contact details.
- Update CV – A client can replace the old CV with a new up to date CV.

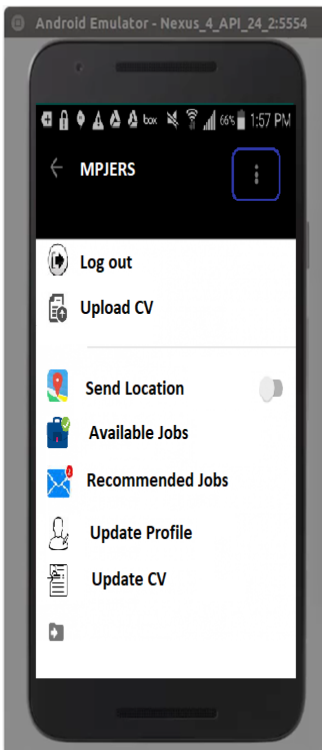


Fig 4.3(a) – Client Service List

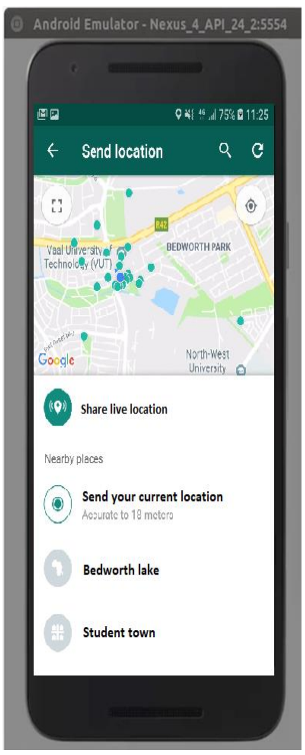


Fig 4.3(b) – Send Location

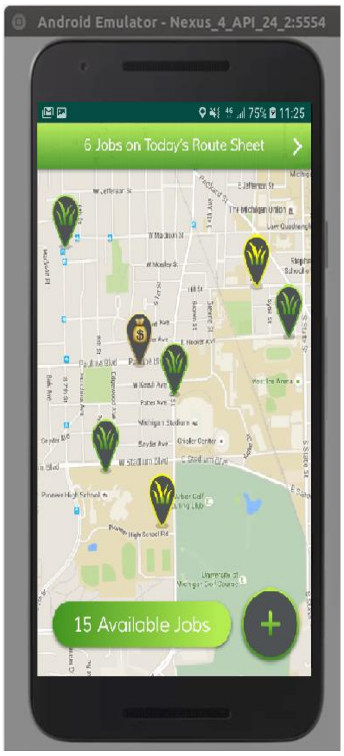

Android Emulator - Nexus_4_API_24_2:5554
75% 11:25
6 Jobs on Today's Route Sheet
15 Available Jobs

Fig 4.3(c) – Available Jobs

Fig 4.3 - MPJERS Client Functionalities

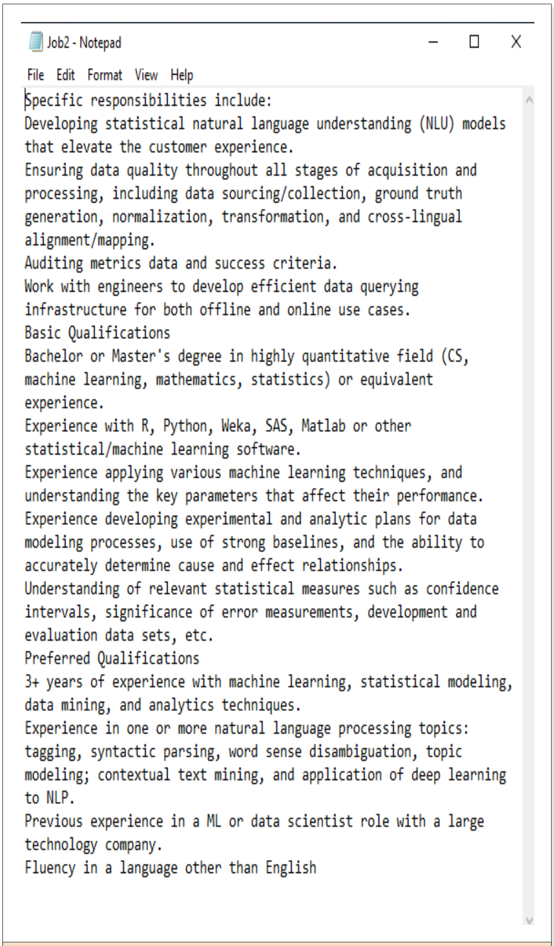
4.7.2 Server side

The server side receives information from the client side and recommends a temporary job to a suitable mobile job seeker. Information stored in SQLite database of the client application is transferred to the server side through the web, where the user profile is generated and stored in MySql database. The resumes from each profile in the database are then collected for recommendation. Firstly, the resumes/CVs and query text documents (job description) are structured in a suitable form for information retrieval using the major preprocessing steps and then grouped into clusters of similar documents regarding the field of job. Secondly the similarities between the job description and resumes were calculated and lastly the proximity of recommended job seekers to the area where the job will be done was calculated to recommend the most suitable job seeker.

4.7.2.1 Text Document Preprocessing

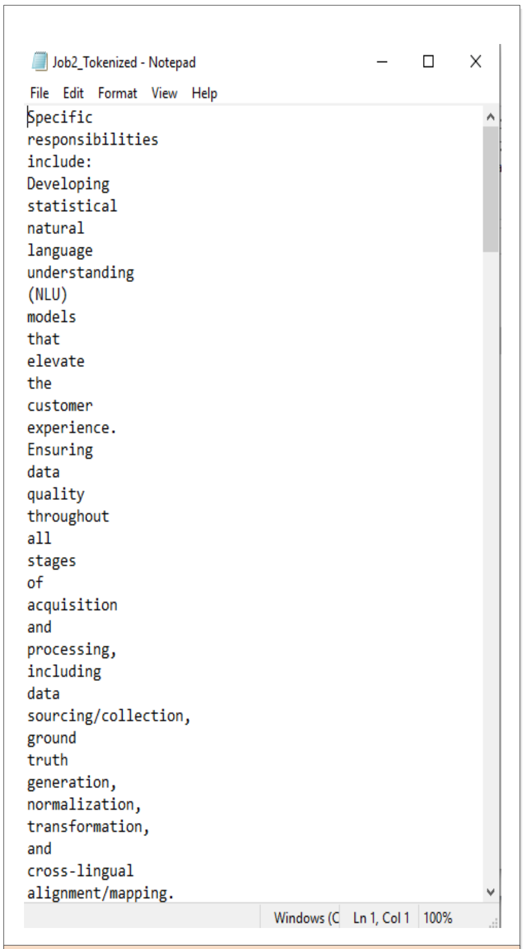
Three major preprocessing steps that were discussed in section 3.5.1 are applied to structure all the text documents.

Fig 4.4 shows a sample of how one of the text documents looked like before and after tokenization is applied on it. Fig 4.4(a) shows an original job description text document before tokenization is applied. Fig 4.4(b) illustrates the resulting job description text document after tokenization is applied. The main objectives of tokenization are to split texts into plain processing units and to interpret and collect isolated tokens to produce higher level tokens.



```
Specific responsibilities include:
Developing statistical natural language understanding (NLU) models
that elevate the customer experience.
Ensuring data quality throughout all stages of acquisition and
processing, including data sourcing/collection, ground truth
generation, normalization, transformation, and cross-lingual
alignment/mapping.
Auditing metrics data and success criteria.
Work with engineers to develop efficient data querying
infrastructure for both offline and online use cases.
Basic Qualifications
Bachelor or Master's degree in highly quantitative field (CS,
machine learning, mathematics, statistics) or equivalent
experience.
Experience with R, Python, Weka, SAS, Matlab or other
statistical/machine learning software.
Experience applying various machine learning techniques, and
understanding the key parameters that affect their performance.
Experience developing experimental and analytic plans for data
modeling processes, use of strong baselines, and the ability to
accurately determine cause and effect relationships.
Understanding of relevant statistical measures such as confidence
intervals, significance of error measurements, development and
evaluation data sets, etc.
Preferred Qualifications
3+ years of experience with machine learning, statistical modeling,
data mining, and analytics techniques.
Experience in one or more natural language processing topics:
tagging, syntactic parsing, word sense disambiguation, topic
modeling; contextual text mining, and application of deep learning
to NLP.
Previous experience in a ML or data scientist role with a large
technology company.
Fluency in a language other than English
```

Fig 4.4(a) – Original Text Document

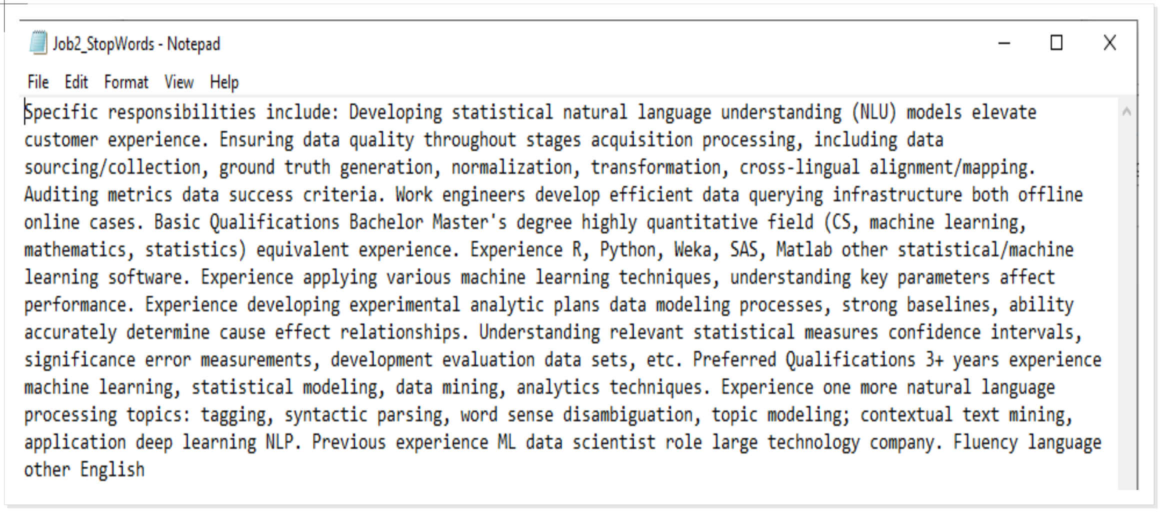


```
Specific
responsibilities
include:
Developing
statistical
natural
language
understanding
(NLU)
models
that
elevate
the
customer
experience.
Ensuring
data
quality
throughout
all
stages
of
acquisition
and
processing,
including
data
sourcing/collection,
ground
truth
generation,
normalization,
transformation,
and
cross-lingual
alignment/mapping.
```

Fig 4.4(b) – Tokenized Text Document

Fig 4.4 - Text Document Tokenization

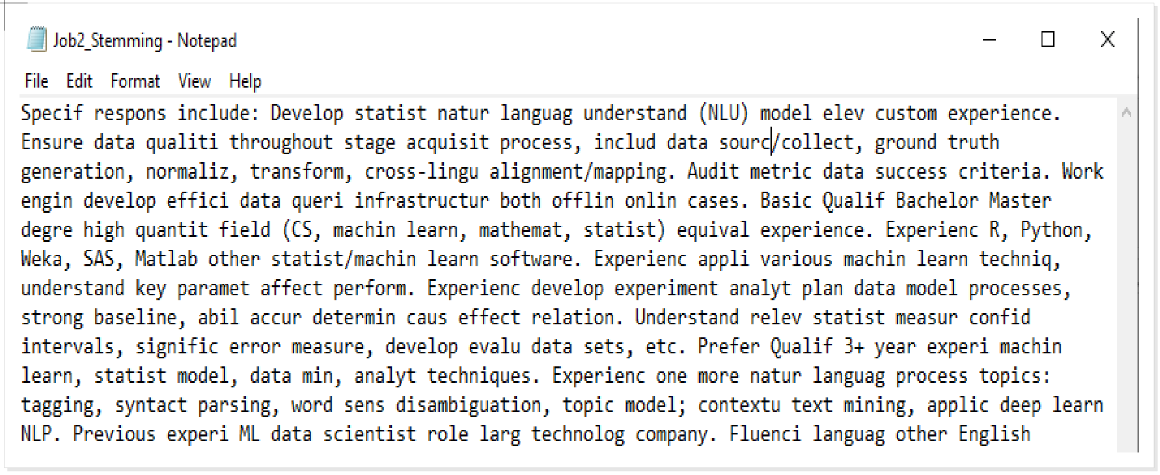
After tokenization of a text document, the next major step in information retrieval preprocessing is the removal of stopwords from a text document. Fig 4.5 illustrates an example of how a tokenized document looked after the stopwords has been removed.



```
Specific responsibilities include: Developing statistical natural language understanding (NLU) models elevate
customer experience. Ensuring data quality throughout stages acquisition processing, including data
sourcing/collection, ground truth generation, normalization, transformation, cross-lingual alignment/mapping.
Auditing metrics data success criteria. Work engineers develop efficient data querying infrastructure both offline
online cases. Basic Qualifications Bachelor Master's degree highly quantitative field (CS, machine learning,
mathematics, statistics) equivalent experience. Experience R, Python, Weka, SAS, Matlab other statistical/machine
learning software. Experience applying various machine learning techniques, understanding key parameters affect
performance. Experience developing experimental analytic plans data modeling processes, strong baselines, ability
accurately determine cause effect relationships. Understanding relevant statistical measures confidence intervals,
significance error measurements, development evaluation data sets, etc. Preferred Qualifications 3+ years experience
machine learning, statistical modeling, data mining, analytics techniques. Experience one more natural language
processing topics: tagging, syntactic parsing, word sense disambiguation, topic modeling; contextual text mining,
application deep learning NLP. Previous experience ML data scientist role large technology company. Fluency language
other English
```

Fig 4.5 - Text document after Stopwords removal

After removal of stopwords, the last major step in information retrieval, stemming, is applied on the documents so that they are ready for information retrieval. The main objective of stemming is to replace all variants of the word with the single stem of the word. Fig 4.6 illustrates the resulting document after stemming is applied on a document. The English snowball library was used to stem the English words from text documents.



```
Specif respons include: Develop statist natur languag understand (NLU) model elev custom experience.
Ensure data qualiti throughout stage acquisit process, includ data sourc/collect, ground truth
generation, normaliz, transform, cross-lingu alignment/mapping. Audit metric data success criteria. Work
engin develop effici data queri infrastructur both offlin onlin cases. Basic Qualif Bachelor Master
degre high quantit field (CS, machin learn, mathemat, statist) equival experience. Experienc R, Python,
Weka, SAS, Matlab other statist/machin learn software. Experienc appli various machin learn techniq,
understand key paramet affect perform. Experienc develop experiment analyt plan data model processes,
strong baseline, abil accur determin caus effect relation. Understand relev statist measur confid
intervals, signific error measure, develop evalu data sets, etc. Prefer Qualif 3+ year experi machin
learn, statist model, data min, analyt techniques. Experienc one more natur languag process topics:
tagging, syntact parsing, word sens disambiguation, topic model; contextu text mining, applic deep learn
NLP. Previous experi ML data scientist role larg technolog company. Fluenci languag other English
```

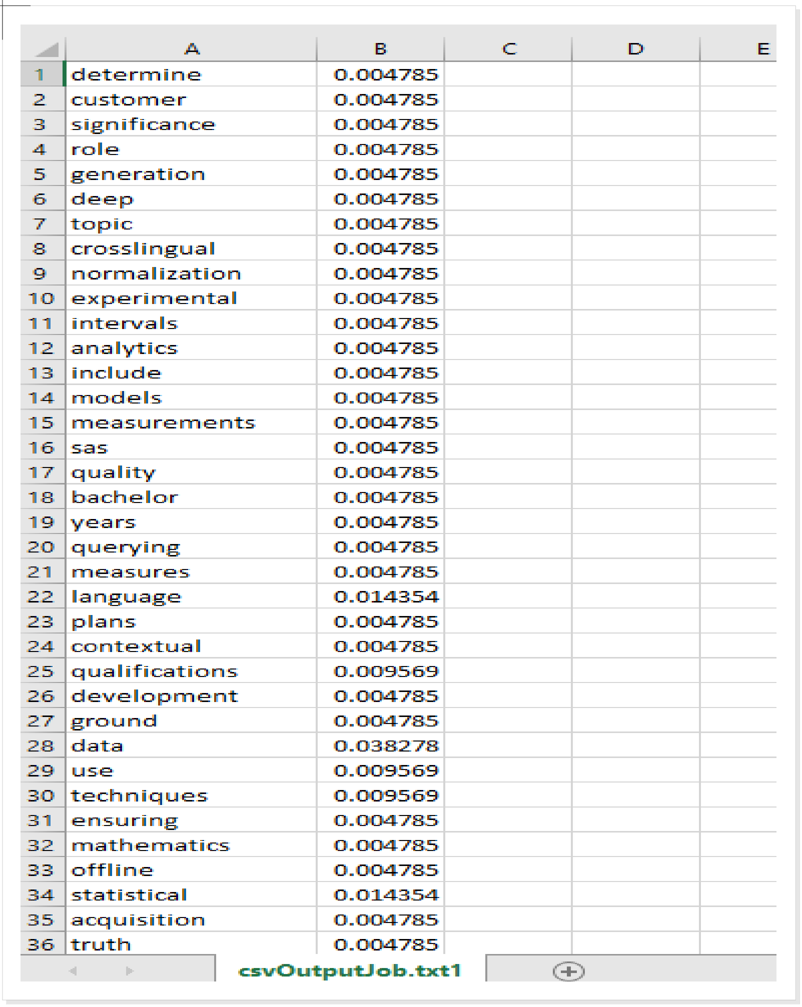
Fig 4.6 - Text document after Stemming

4.7.2.2 Text document clustering

After the major preprocessing techniques have been applied on all resumes and job descriptions text document to remove noise from text documents, the root words can then be easily identified. Resumes text documents are now ready to be compared for similarity and the performance for recommending the most similar resumes to the queries (job description text documents) is improved.

4.7.2.3 Similarity

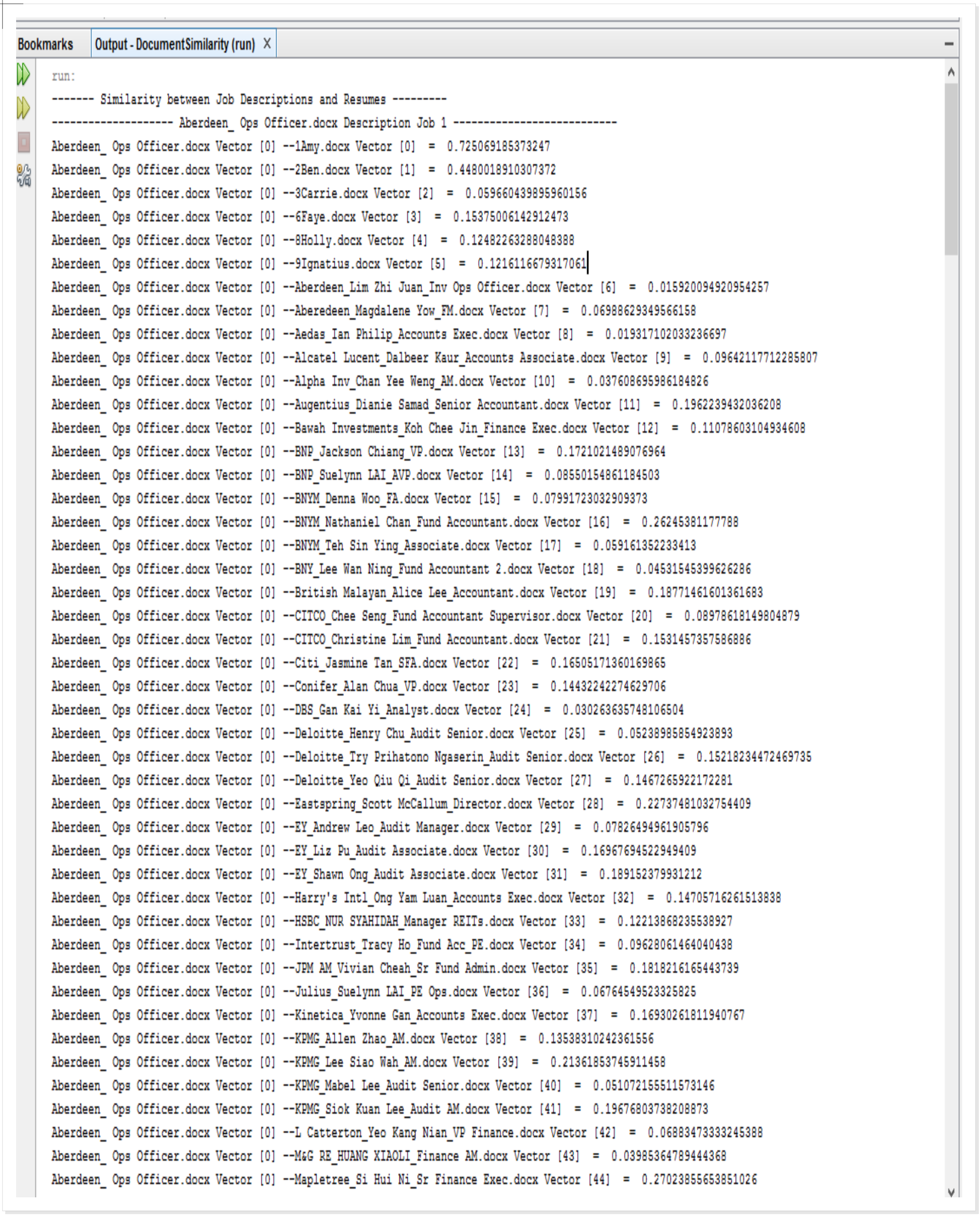
To find the similarity between the resumes and queries documents, both sets of documents are now represented as vectors of terms. The next step is to represent the vector of terms to numerical format known as term document matrix as explained in section 3.5.2. After the term document matrix is generated, the term weights of each term in the matrix across all the documents are calculated. It is important to compute the term weights because of the need to identify terms which uniquely define each document. In this research study, TF-IDF was used to calculate the term weights. Fig 4.7 shows the screenshot of term weighting from one document.



	A	B	C	D	E
1	determine	0.004785			
2	customer	0.004785			
3	significance	0.004785			
4	role	0.004785			
5	generation	0.004785			
6	deep	0.004785			
7	topic	0.004785			
8	crosslingual	0.004785			
9	normalization	0.004785			
10	experimental	0.004785			
11	intervals	0.004785			
12	analytics	0.004785			
13	include	0.004785			
14	models	0.004785			
15	measurements	0.004785			
16	sas	0.004785			
17	quality	0.004785			
18	bachelor	0.004785			
19	years	0.004785			
20	querying	0.004785			
21	measures	0.004785			
22	language	0.014354			
23	plans	0.004785			
24	contextual	0.004785			
25	qualifications	0.009569			
26	development	0.004785			
27	ground	0.004785			
28	data	0.038278			
29	use	0.009569			
30	techniques	0.009569			
31	ensuring	0.004785			
32	mathematics	0.004785			
33	offline	0.004785			
34	statistical	0.014354			
35	acquisition	0.004785			
36	truth	0.004785			

Fig 4.7 - Term Weight Results

After the term weights are calculated, the similarity between the job description and the resumes were calculated using cosine similarity as in equation 22 in section 3.7. Fig 4.8 displays the similarities between job description text documents and resumes.

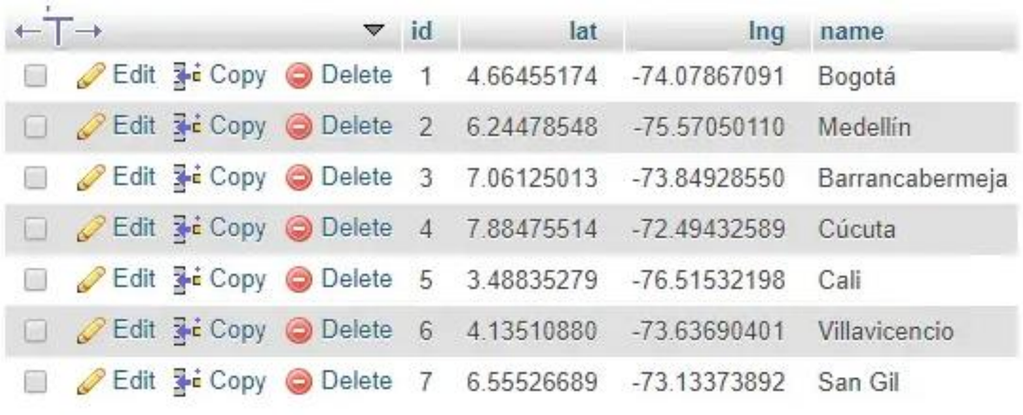


```
run:
------- Similarity between Job Descriptions and Resumes ---------
------------------- Aberdeen_ Ops Officer.docx Description Job 1 --------------------------
Aberdeen_ Ops Officer.docx Vector [0] --1Amy.docx Vector [0]  =  0.725069185373247
Aberdeen_ Ops Officer.docx Vector [0] --2Ben.docx Vector [1]  =  0.4480018910307372
Aberdeen_ Ops Officer.docx Vector [0] --3Carrie.docx Vector [2]  =  0.059660439895960156
Aberdeen_ Ops Officer.docx Vector [0] --6Faye.docx Vector [3]  =  0.15375006142912473
Aberdeen_ Ops Officer.docx Vector [0] --8Holly.docx Vector [4]  =  0.12482263288048388
Aberdeen_ Ops Officer.docx Vector [0] --9Ignatius.docx Vector [5]  =  0.1216116679317061
Aberdeen_ Ops Officer.docx Vector [0] --Aberdeen_Lim Zhi Juan_Inv Ops Officer.docx Vector [6]  =  0.015920094920954257
Aberdeen_ Ops Officer.docx Vector [0] --Aberedeen_Magdalene Yow_FM.docx Vector [7]  =  0.06988629349566158
Aberdeen_ Ops Officer.docx Vector [0] --Aedas_Ian Philip_Accounts Exec.docx Vector [8]  =  0.019317102033236697
Aberdeen_ Ops Officer.docx Vector [0] --Alcatel Lucent_Dalbeer Kaur_Accounts Associate.docx Vector [9]  =  0.09642117712285807
Aberdeen_ Ops Officer.docx Vector [0] --Alpha Inv_Chan Yee Weng_AM.docx Vector [10]  =  0.037608695986184826
Aberdeen_ Ops Officer.docx Vector [0] --Augentius_Dianie Samad_Senior Accountant.docx Vector [11]  =  0.1962239432036208
Aberdeen_ Ops Officer.docx Vector [0] --Bawah Investments_Koh Chee Jin_Finance Exec.docx Vector [12]  =  0.11078603104934608
Aberdeen_ Ops Officer.docx Vector [0] --BNP_Jackson Chiang_VP.docx Vector [13]  =  0.1721021489076964
Aberdeen_ Ops Officer.docx Vector [0] --BNP_Suelynn LAI_AVP.docx Vector [14]  =  0.08550154861184503
Aberdeen_ Ops Officer.docx Vector [0] --BNYM_Denna Woo_FA.docx Vector [15]  =  0.07991723032909373
Aberdeen_ Ops Officer.docx Vector [0] --BNYM_Nathaniel Chan_Fund Accountant.docx Vector [16]  =  0.26245381177788
Aberdeen_ Ops Officer.docx Vector [0] --BNYM_Teh Sin Ying_Associate.docx Vector [17]  =  0.059161352233413
Aberdeen_ Ops Officer.docx Vector [0] --BNY_Lee Wan Ning_Fund Accountant 2.docx Vector [18]  =  0.04531545399626286
Aberdeen_ Ops Officer.docx Vector [0] --British Malayan_Alice Lee_Accountant.docx Vector [19]  =  0.18771461601361683
Aberdeen_ Ops Officer.docx Vector [0] --CITCO_Chee Seng_Fund Accountant Supervisor.docx Vector [20]  =  0.08978618149804879
Aberdeen_ Ops Officer.docx Vector [0] --CITCO_Christine Lim_Fund Accountant.docx Vector [21]  =  0.1531457357586886
Aberdeen_ Ops Officer.docx Vector [0] --Citi_Jasmine Tan_SFA.docx Vector [22]  =  0.16505171360169865
Aberdeen_ Ops Officer.docx Vector [0] --Conifer_Alan Chua_VP.docx Vector [23]  =  0.14432242274629706
Aberdeen_ Ops Officer.docx Vector [0] --DBS_Gan Kai Yi_Analyst.docx Vector [24]  =  0.030263635748106504
Aberdeen_ Ops Officer.docx Vector [0] --Deloitte_Henry Chu_Audit Senior.docx Vector [25]  =  0.05238985854923893
Aberdeen_ Ops Officer.docx Vector [0] --Deloitte_Try Prihatono Ngaserin_Audit Senior.docx Vector [26]  =  0.15218234472469735
Aberdeen_ Ops Officer.docx Vector [0] --Deloitte_Yeo Qiu Qi_Audit Senior.docx Vector [27]  =  0.146726592217228l
Aberdeen_ Ops Officer.docx Vector [0] --Eastspring_Scott McCallum_Director.docx Vector [28]  =  0.22737481032754409
Aberdeen_ Ops Officer.docx Vector [0] --EY_Andrew Leo_Audit Manager.docx Vector [29]  =  0.07826494961905796
Aberdeen_ Ops Officer.docx Vector [0] --EY_Liz Pu_Audit Associate.docx Vector [30]  =  0.16967694522949409
Aberdeen_ Ops Officer.docx Vector [0] --EY_Shawn Ong_Audit Associate.docx Vector [31]  =  0.189152379931212
Aberdeen_ Ops Officer.docx Vector [0] --Harry's Intl_Ong Yam Luan_Accounts Exec.docx Vector [32]  =  0.14705716261513838
Aberdeen_ Ops Officer.docx Vector [0] --HSBC_NUR SYAHIDAH_Manager REITs.docx Vector [33]  =  0.12213868235538927
Aberdeen_ Ops Officer.docx Vector [0] --Intertrust_Tracy Ho_Fund Acc_PE.docx Vector [34]  =  0.09628061464040438
Aberdeen_ Ops Officer.docx Vector [0] --JPM AM_Vivian Cheah_Sr Fund Admin.docx Vector [35]  =  0.181821616544373 9
Aberdeen_ Ops Officer.docx Vector [0] --Julius_Suelynn LAI_PE Ops.docx Vector [36]  =  0.06764549523325825
Aberdeen_ Ops Officer.docx Vector [0] --Kinetica_Yvonne Gan_Accounts Exec.docx Vector [37]  =  0.16930261811940767
Aberdeen_ Ops Officer.docx Vector [0] --KPMG_Allen Zhao_AM.docx Vector [38]  =  0.13538310242361556
Aberdeen_ Ops Officer.docx Vector [0] --KPMG_Lee Siao Wah_AM.docx Vector [39]  =  0.21361853745911458
Aberdeen_ Ops Officer.docx Vector [0] --KPMG_Mabel Lee_Audit Senior.docx Vector [40]  =  0.051072155511573146
Aberdeen_ Ops Officer.docx Vector [0] --KPMG_Siok Kuan Lee_Audit AM.docx Vector [41]  =  0.19676803738208873
Aberdeen_ Ops Officer.docx Vector [0] --L Catterton_Yeo Kang Nian_VP Finance.docx Vector [42]  =  0.06883473333245388
Aberdeen_ Ops Officer.docx Vector [0] --M&G RE_HUANG XIAOLI_Finance AM.docx Vector [43]  =  0.03985364789444368
Aberdeen_ Ops Officer.docx Vector [0] --Mapletree_Si Hui Ni_Sr Finance Exec.docx Vector [44]  =  0.27023855653851026
```

Fig 4.8 - Document Similarity

4.7.2.4 Proximity

0nce the most relevant resumes have been recommended, the proximity of recommended job seekers to the area where the job will be done is calculated. The task of the location manager is to locate the nearest recommended job seeker to the area where the job will be done given the current position coordinates (latitude and longitude) of the job seeker. The markers are used on the map to store job seekers locations and the coordinates are stored in MySQL database. The database stores markers that are identified by a name and contains coordinates of latitude and longitude as illustrated in Fig 4.9.



				id	lat	lng	name
	Edit	Copy	Delete	1	4.66455174	-74.07867091	Bogotá
	Edit	Copy	Delete	2	6.24478548	-75.57050110	Medellín
	Edit	Copy	Delete	3	7.06125013	-73.84928550	Barrancabermeja
	Edit	Copy	Delete	4	7.88475514	-72.49432589	Cúcuta
	Edit	Copy	Delete	5	3.48835279	-76.51532198	Cali
	Edit	Copy	Delete	6	4.13510880	-73.63690401	Villavicencio
	Edit	Copy	Delete	7	6.55526689	-73.13373892	San Gil

Fig 4.9 - Screenshot of Coordinates and Location from MySQL Database

After placing the marker of the locations in Google Maps using JavaScript, the output on the map will be as illustrated in Fig 4.10. Flag markers on the map represent current locations of recommended job candidates and a pin maker represents the place where the job will be done.

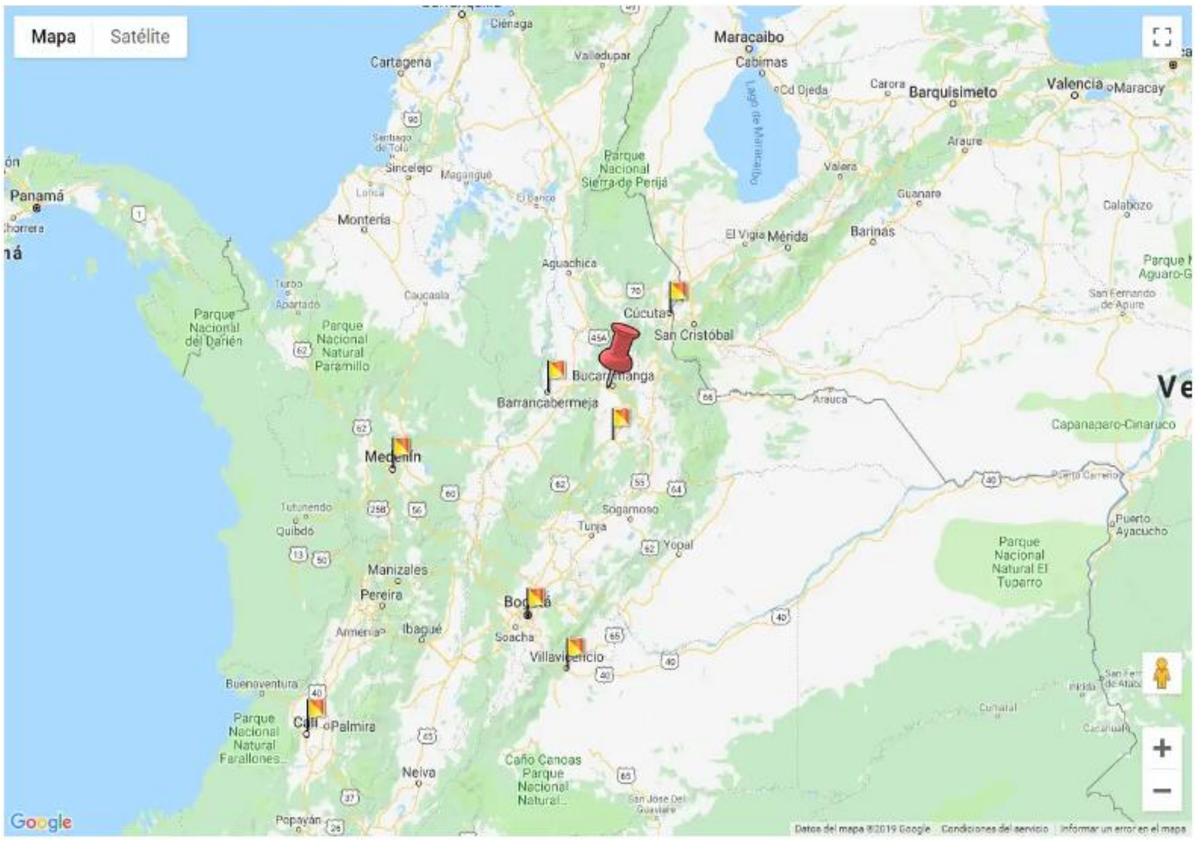


Fig 4.10 - Marker of Locations in Google Maps

Now that the markers are in the correct location, it is important to filter them according to the distance from the area where the job will be done. The distance is determined through a circular area, defined by some distance that is named search radius as illustrated in Fig 4.11.

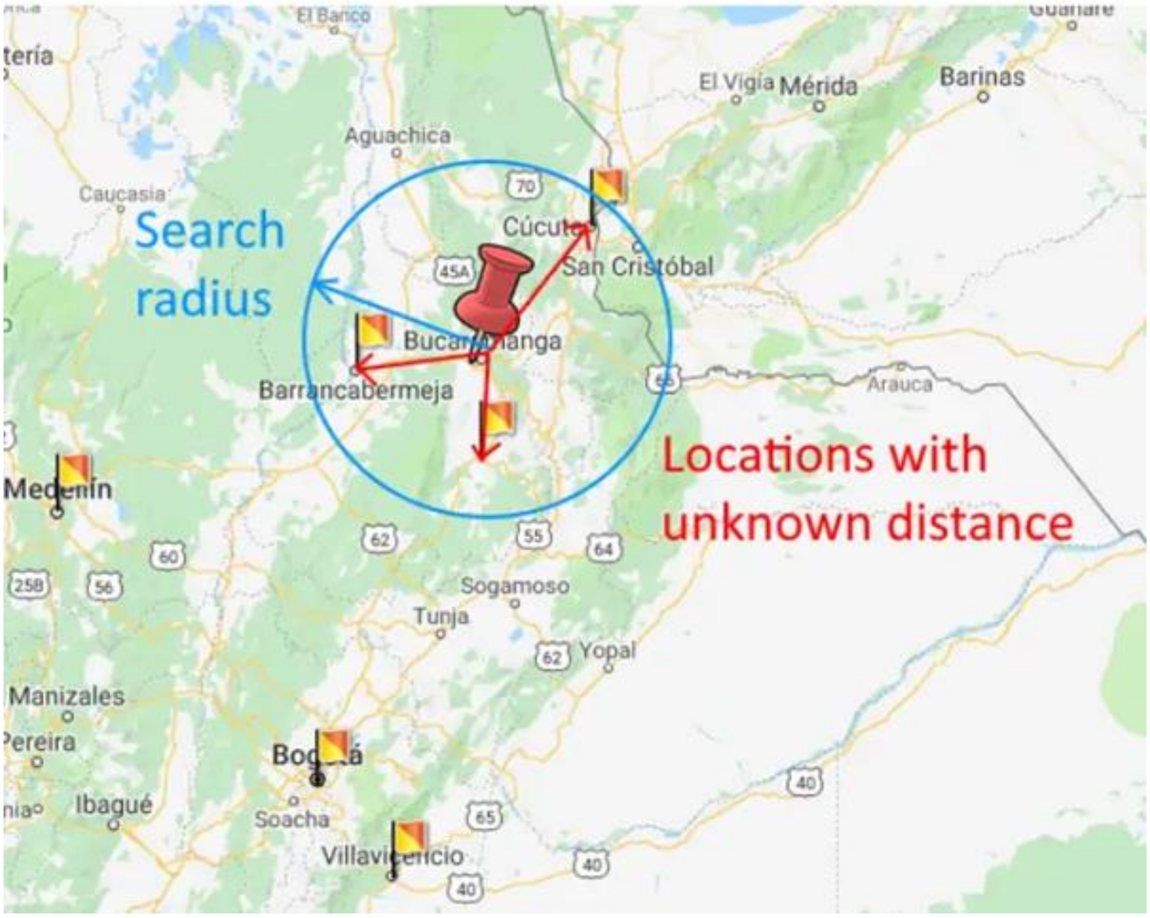
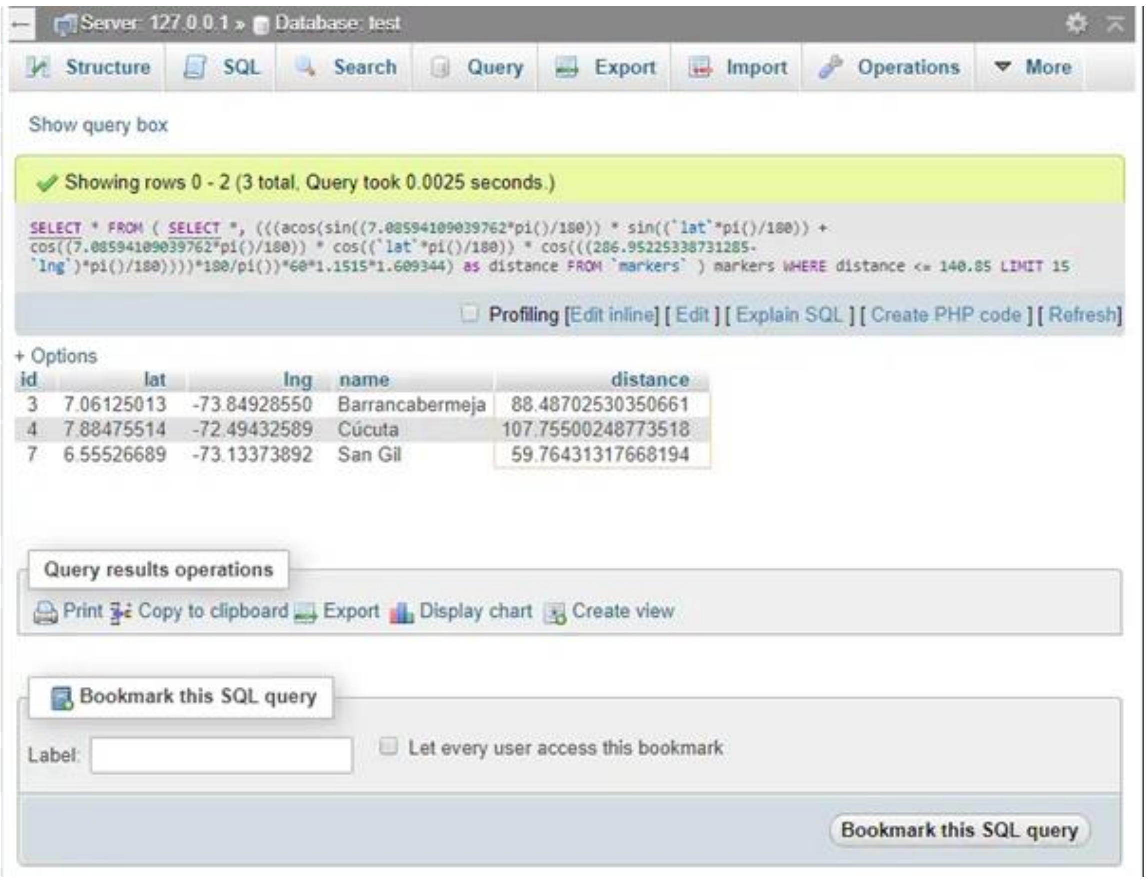


Fig 4.11 – Recommended Search Radius

To define the search radius in Fig 4.11, the initial point was the area where the job was going to be performed. The distance of 110 kilometers from the initial point was used as the radius and the circular area was generated. To find the locations inside the circular area, equation 16 in section 3.6 was used and recommended is shown in Fig 4.12.



Showing rows 0 - 2 (3 total, Query took 0.0025 seconds.)

```
SELECT * FROM ( SELECT *, (((acos(sin((7.08594109039762*pi()/180)) * sin((`lat`*pi()/180)) +
cos((7.08594109039762*pi()/180)) * cos((`lat`*pi()/180)) * cos(((286.95225338731285-
`lng`)*pi()/180))))*180/pi())*60*1.1515*1.609344) as distance FROM `markers` ) markers WHERE distance <= 140.85 LIMIT 15
```

id	lat	lng	name	distance
3	7.06125013	-73.84928550	Barrancabermeja	88.48702530350661
4	7.88475514	-72.49432589	Cúcuta	107.75500248773518
7	6.55526689	-73.13373892	San Gil	59.76431317668194

Fig 4.12 - Recommended Locations

4.8 CHAPTER SUMMARY

This chapter presented how the prototype of the recommender system was implemented and the results that the prototype achieved. Firstly, the requirements that the system need to achieve were classified into functional and non – functional requirements. Next, the programming environments that were used for building the prototype were explained. Likewise, various aspects of the GPS were explained.

Furthermore, the proposed content-based recommender engine was clarified. Datasets that were used to test the prototype were also demonstrated. Lastly, the prototype implementation and results were clearly demonstrated. This was initiated from the implementation of the mobile application and finalized by recommendation of suitable job seekers that were closest to the area where the job was to be done. The following chapter will elaborate on how the system was evaluated.

CHAPTER 5: SYSTEM EVALUATION

5.1 INTRODUCTION

Since the prototype of the MPJERS is a client server architecture, the recommender system was evaluated on both the client and server side. In this chapter, the focus is on how the recommender system was evaluated. Both the client side and the server side were evaluated for performance.

The evaluation of the recommender system was done offline. Offline experiments are performed using pre-collected datasets. The datasets are used to simulate the behaviour of users that interact with a recommender system while there is no real interaction with users. The datasets used in this study were obtained from Kaggle website (www.kaggle.com).

5.2 CLIENT-SIDE EVALUATION

In regards to evaluating the client side's performance, perfecto mobile testing tool was used to check the performance of the developed mobile application. The next section elaborates on how performance of a mobile application was evaluated.

5.2.1 Mobile application performance

So as to assess the performance of the mobile application in the client side, we utilized Perfecto mobile which is one of android debugging tool. Debug checks are a fundamental piece of the application and they can screen an application that is running, giving bits of knowledge to the exhibition of the application through its utilization of system resources. To assess the developed mobile application, we ran it on a real android gadget and we ran the functionalities that are offered by the application, such as generating a user profile, uploading a CV, viewing available jobs and so on.

The initial form of debug check that was inspected was about the gadget's CPU. Fig 5.1 illustrates a historic the CPU's utilization graph over the time the application was being utilized.

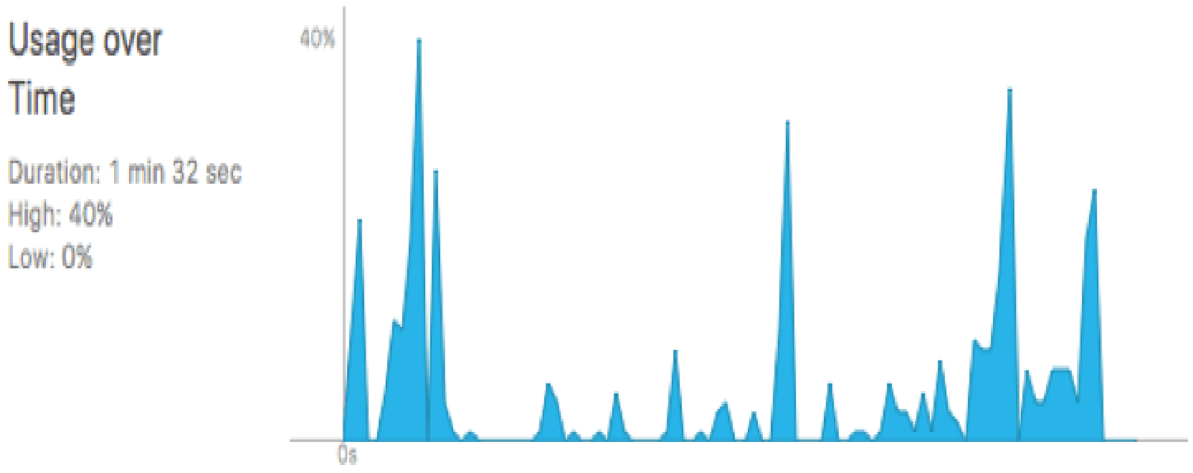


Fig 5.1 - CPU Utilization Graph

Fig 5.1 indicates that the most elevated level of CPU utilization was 40% and, as demonstrated on the meter view in Fig 5.2, CPU usage could reach up to 200%, since the android gadget that was utilized had a dual-core processor.

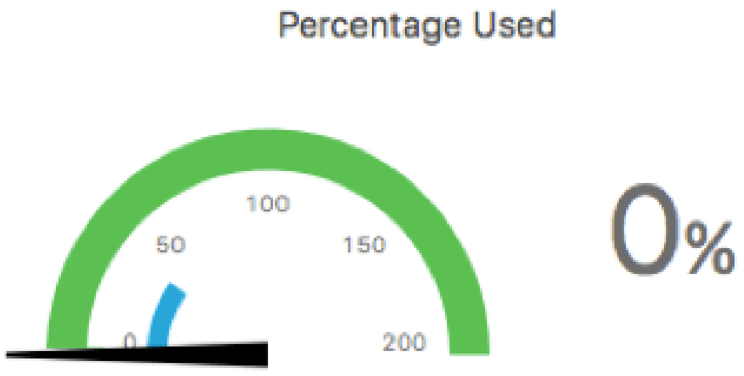


Fig 5.2 - CPU utilization meter view

Through the memory check, there was an option to verify that the application expected to utilize just 21.8 MB, which comprises the 1.1% of the overall system memory. The outcomes in regards to the memory utilization can be found in the subsequent meter view.

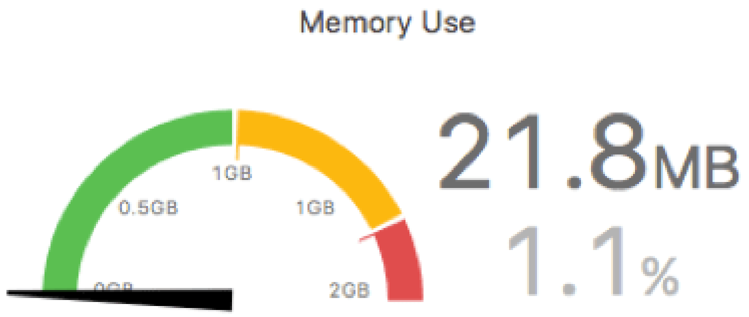


Fig 5.3 - Memory utilization meter view

With regards to vitality effect of the developed mobile application, the pie chart in Fig 5.4 shows the normal vitality usage by the various components. It may very well be seen that the overhead is liable for most of the vitality that is being utilized. This can be simply clarified by the way that the application performs numerous activities so as to send and get information from the system's database and such interchanges frequently lead to broad overhead cost.

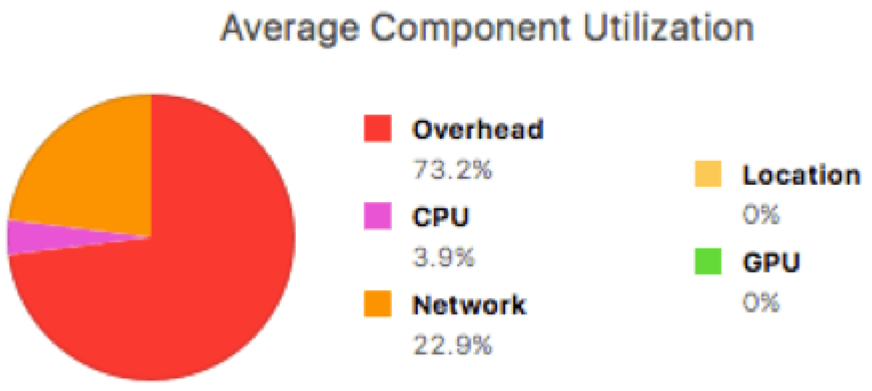


Fig 5.4 - Vitality usage pie chart

At last, in regards to the system activity all through the performance of the previously mentioned operations of the mobile application, we saw that 0.4 MB were found by the device and just 11.2 KB were sent by it.

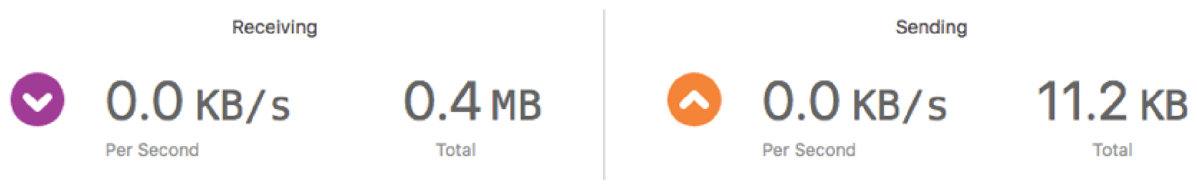


Fig 5.5 - System activity

5.3 SERVER-SIDE EVALUATION

The server side of the MPJERS was evaluated for performance and accuracy. Performance measures of precision and recall were used to evaluate the document classification while accuracy measure was used evaluate the location correctness. The next section presents the details of both document classification and location correctness.

5.3.1 Document Classification

To evaluate the text classification algorithm, the results of the real output were compared with the results of the classification process. The metrics used were the number of True Positives, True Negatives, False Positives and False Negatives. In this setting, True Positives measure the precision of the system to detect if the current document is one of the relevant documents to be classified. A True Negative measure how good the system detects non-standard documents and classify these documents as “Other”. False Positive measures if the system fails to classify a document that is supposed to be labelled as “Others” and instead is labelled as one of the relevant documents. On the other hand, a False Negative occurs when a relevant document is tagged as “Others” and therefore misclassified.

Table 5.1 – Document Classification results

System Classification			
Relevant	Other		
1395	0	Relevant	Real Classification
8	36	Other	

As it can be seen in Table 5.1, the number of relevant documents that were acceptably classified is 1395 from 1431. Also, all the documents that were supposed to be categorized as others were classified properly as well. Only eight documents were incorrectly classified as “Others”. This is because some of the document were scanned images not text documents making the system not able to fail to extract data from them. To evaluate the document classifications, precision, recall and F1 scores were calculated as shown in section 3.9. Firstly, precision was calculated using equation 24 and the precision value was obtained as 0.994. Then the recall value was obtained as 0.975 after applying equation 25. Finally, a single metric that combines precision and recall using the harmonic mean, the F1-Score of the classification system was obtained as 0.984 and was calculated by applying equation 26.

Table 5.2 – Evaluation Metrics

Metric	Result
Precision	1395/(1395+8) = 0.994
Recall	1395/(1395+36) = 0.975
F1 – Score (relates precision and recall) $\frac{2 * 0.994 * 0.975}{0.994 + 0.975} = 0.984$	

Table 5.2 uses the data from Table 5.1 to calculate the metrics that help to evaluate recommendation obtained in a more precise way. These metrics can then be analyzed. Analyzing the calculated metrics, it can be concluded that the performance of the MPJERS is of very high quality as illustrated by the precision and recall curve in Fig 5.6.

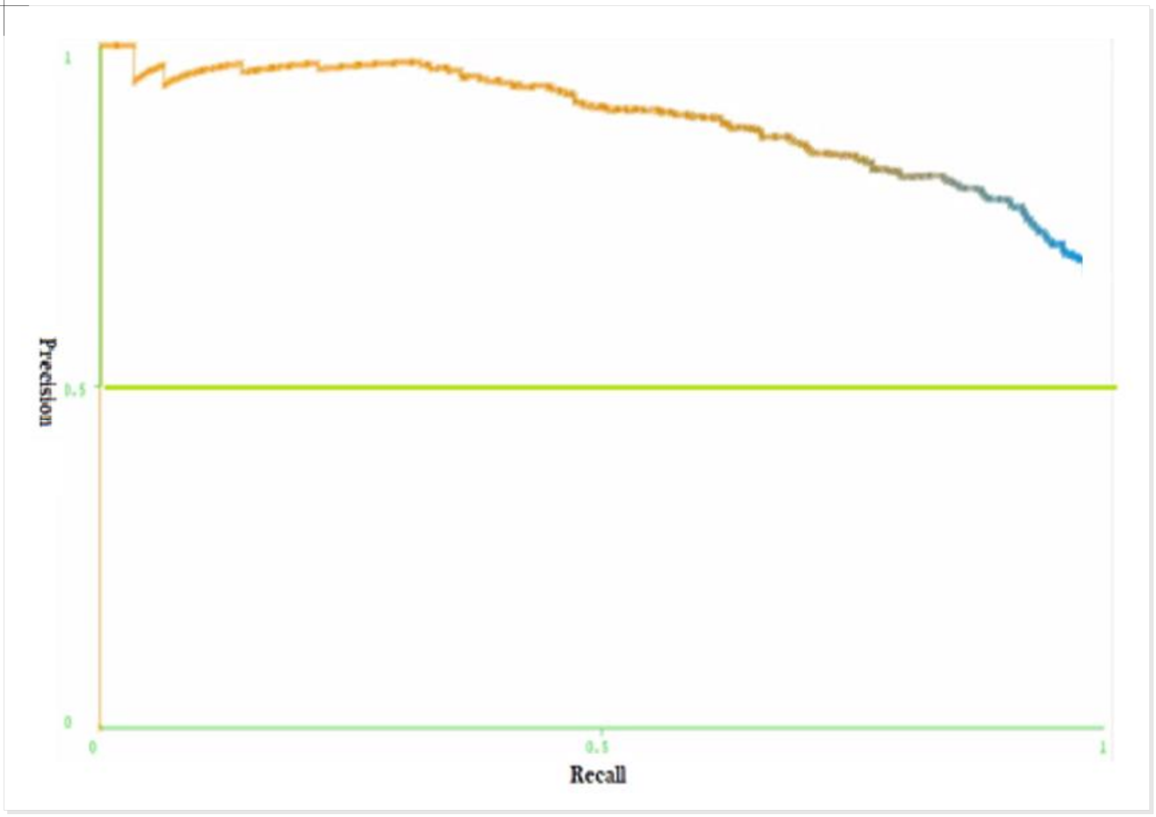


Fig 5.6 - Precision and Recall curve

5.4 CHAPTER SUMMARY

This chapter presented how the proposed recommender system was evaluated. The proposed system prototype uses a client – server architecture, thus, the system was evaluated on both the client and the server side. The perfecto mobile testing tool was used to test the mobile application's performance in the client side. On the other hand, precision and recall were used as the measure of performance in the server side. The next chapter will give the conclusion and suggestions for the what can be improved in the system.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 INTRODUCTION

This chapter outlines the conclusion and future work after completion of this research study. The conclusion is derived on whether the research was able to achieve its aim and objectives and the future work outlines the suggestions that can help improve the system.

6.2 CONCLUSION

This research project proposed a temporary job employment recommender system for mobile users. Recommendation for a job is separated into two parts: Initially, the similarity between job seekers' resume text documents and recruiters job description text documents is calculated using cosine similarity and resumes with highest similarity scores are shortlisted. Lastly, shortlisted job seekers proximity to the area where the job will be done is checked and nearest job seeker to the place of work is recommended for the temporary job.

To recommend the most appropriate resumes, both resumes and job description text documents were structured into a suitable format for information retrieval using TDF-IDF and the major preprocessing techniques. After preprocessing of text documents, resume documents are grouped into different clusters of similar documents. Finally, similarity between job description and resumes was calculated a shortlist of recommendation was done.

To check the proximity of shortlisted mobile job seekers to the area where the job will be done, the current location of the mobile job seekers is obtained through the location manager. The longitude and latitude coordinates of both job seekers' current position and the area where the job will be done are then displayed on Google maps. Markers are used to show the coordinates of shortlisted candidates and the distance from the place of work is determined by the search radius from the place of work. The closest candidate to the place of work are then recommended for a specified temporary job.

In order to evaluate the performance and effectiveness of the system, the evaluation metrices of precision and recall were measured on the resume text documents. The value of precision was found to be 0.994 and recall value was 0.975. To combine precision and recall, the F1 measure (harmonic mean) was calculated and the value of 0.984 was found. These results implied that the system performance is very impressive and hence the system is effective as all the results were correct. The system was able to achieve all the functional and non-functional requirements that were set in section 4.2.

6.3 FUTURE WORK

Even though the proposed MPJERS could answer the research question in this dissertation, it is clear additional work could still be done to extend the functionalities and further evaluate the recommender system, while also adding more value to the to the developed system.

A hybrid recommender system that combines content–based and collaborative filtering algorithm can be developed to reduce the problems related in using only content-based. These problems may include the sparsity problems that can occur when adding new users in the system.

Furthermore, to evaluate the prototype more precisely, the actual users' feedback will be very useful. It would help to generate a better user experience to the system and the recommendation criteria of the system can be expanded based on the users' needs. Also, the actual user interaction with the mobile application would help to evaluate the scalability of the system, more especially if it is going to be used in real world environment.

REFERENCES

ABUALIGAH, L.M., KHADER, A.T. & HENANDEH, E.S. 2018. A Hybrid Strategy for Krill Herd Algorithm with Harmony search Algorithm to Improve the Data Clustering: Article in Intelligent Decision Technologies. *ResearchGate*. 1 – 12.

ADOMAVICIUS, G. & KWON, Y. 2012. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Transactions on Knowledge and Data Engineering*. 24(5): 896 – 911.

ADOMAVICIUS G., TUZHILIN A. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State of the Art and ossible Extensions. *IEEE Transactions on Knowledge and Data Engineering*. 17: 734 - 749.

ALEXANDER F., KLAUS I., KALMAN S., & PETER Z. 2007. The VITA Financial Services Sales Support Environment. *AAAI*. 1692 - 1699.

AL-OTAIBI, S. T. & YKHIEF, M. 2017. Hybrid immunizing solution for job recommender system. *Frontiers of Computer Science*. 11: 511 – 527.

AL-OTAIBI, S.T. & YKHIEF, M. 2012. Recommendation Systems for Enhancing E-recruitment Process. *Proceedings of the International Conference on Information and Knowledge Engineering.*

ANKERST, M., BREUNIG M. M., PETER KRIEGEL H., & SANDER, J. 1999. Ordering points to identify the clustering structure. *ACM Press*. 49 – 60.

ARORA, G., KUMAR, A., DEVRE, G. S. & GHUMARE, A. 2014. Movie recommendation system based on userssimilarity. *International Journal of Computer Science and Mobile Computing*. 3(4): 765 – 770.

BALTRUNAS, L., LUDWIG, B., PEER, S. & RICCI, F. 2012. Context Relevance Assessment and Exploitation in Mobile Recommender Systems. *Personal and Ubiquitous Computing.* 16: 507 – 526.

BAO, J., ZHENG, Y., WILKIE, D. & MOKBEL, M.F. 2013. A Survey on Recommendations in Location Based Social Networks. *Geoinformatica.* 19: 225 – 265.

BERKHIN, P. 2002. A Survey of clustering data mining techniques. In: *Grouping Multidimensional Data*. Berlin: Springer. 25 -71.

BONEH, D., MICHALEVSKY, Y., NAKIBLY, G., SCHULMAN, A. & GUIRAN, A. 2015. Powerspy: Location tracking using mobile device power analysis. In: *Proceeding 24th USENIX Security Symposium*. Washiton D.C. 785 – 800.

BRANTS, T., CHEN, F. & FARAHAT, A. 2003. A System for New Event Detection. In: *Proceeding of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York: ACM. 330 - 337.

BURKE, R. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*. 12(4): 331–370.

BURKE, R. 2007. Hybrid Web Recommender Systems. The Adaptive Web. Methods and Strategies of Web Personalization. Berlin: Springer. 4321: 377-407.

CEOLHO, B., COSTA, F. & GONCALVES, G.M. 2015. Hyred Hybrid Job Recommendation System. In: *Proceeding 12th International Joint Conference on e-Business and Telecommunications (ICETE)*. Colmar: IEEE.

CHANG, Y., QIAO, Y., JIAN, J. &, X. WANG, H. XIA. 2015. Using Multiple Barometers to Detect the Floor Location of Smart Phones With Built-In Barometric Sensors for Indoor Positioning. IEEE Sensors. 15(4): 7857–7877.

CHANG, G., TAN, C., LI, G & ZHU. C. 2010. Developing Applications on the Android Platform. *Mobile Multimedia Processing.* Berlin: Springer. 5960: 264 – 286.

CHON, J. & CHA, H. 2011. Lifemap: A Smartphone-Based Context Provider for Location-Based Services. *IEEE Pervasive Computing*. 10(2): 58–67.

DAVIDSSON, C. 2010. *Mobile Application Recommender System*. Uppsala University: Sweden.

DEFAYS, D. 1997. An Efficient Algorithm for a Complete Link Method. *The Computer Journal*. Oxford: Oxford Academic. 20(4): 364 – 366.

DESCIOLI, P., KURZBAN, R., KOCH, E. N. & LIBEN-NOWELL, D. 2011. Best Friends Alliances: Friend Ranking, and the Myspace Social Network. *Perspectives on Psychological Science.* 6(1): 6 – 8.

DUNLOP, M., MORRISON, A., MCCALLUM, S., PTASKINSKI, P., RISBEY, C. & STEWART, F. 2004. Focussed Palmtop Information Access through Starfield Displays and Profile Matching. In: *Proceedings of Workshop on Mobile and Ubiquitous Information Access.* Berlin: Springer. 2054: 79-89.

ESTER M., PETER KRIEGEL H., AND XU, X. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases With Noise. In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*. Portland: AAAI Press. 226 - 231.

FARBER, F., WEITZEL T. & KEIM, T. 2003. An Automated Recommendation Approach to Selection in Personnel Recruitment. In: *Proceedings of the 9th Americas Conference on Information Systems.* Tampa: AISeL. 2329 – 2339.

FAZEL-ZARANDI, M & FOX, M.S. 2010. Semantic Matchmaking for Job Recruitment: An Ontolgy Based Hybrid Approach. In: *Proceedings of the 3rd International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web at the 8th International Semantic Web Conference*. Washington D.C.

FUNG, B. C., WANG, K. & ESTER, M. 2003. Hierarchical Document Clustering Using Frequent Item Sets. In: *Proceeding Siam International Conference On Data Mining.* San Fransisco. 59 – 70.

GAO, H., TANG, J., HU, X. & LIU, H. 2015. Content-Aware Point of Interest Recommendation on Location-Based Social Networks. In: *Proceedinds of the 29th AAAI Conference on Artificial Intellegence.* AAAI Press. 1721–1727.

GAVALAS, D. & ECONOMOU, D. 2011. Development Platforms for Mobile Applications Status and Trends. Colmar: IEEE. 28(1)**:** 77-89.

HARTIGAN, J.A. AND WONG, M.A. 1979. A K-means clustering algorithm. *Applied Statistics.* New Jersey: Wiley. 28(1): 100 - 108.

HERLOCKER, J.L. KONSTAN, J. A. BORCHERS, A & RIEDL, J. 1999. An Algorithmic Framework for Performing Collaborative Filtering. In: *Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval*. Berkeley: ACM. 230 – 237.

HONG, W., ZHENG, S. & WANG, H. 2013. A Job Recommender System Based on User Clustering. *Journal of Computers*. 8(8): 1960 – 1967.

ISHAK, I.S. & ALIAS, R.A. 2005. Designing a strategic information system planning methodology For Malaysian institutes of higher learning *(ISP-IPTA)*. *Issues in Information Systems*. Universiti Teknologi Malaysia. 6(1): 326 – 331.

JANNACH, D., ZANKER, M. & FRIEDRICH, G. 2013. Tutorial: Recommender Systems. *International Joint Conference on Artificial Intelligence*. Beijing: China.

XU, J.S. & Wang, L. 2005. *Tcblht:* A New Method of Hierarchical Text Clustering. TCBLHT. 2005 *International Conference on Machine Learning and Cybernetics*. Guangzhou: China. 4: 2178 – 2181.

JAIN, A., CHAUHAN, N., SINGH, V. & THAKUR, N. 2017. Information Retrieval using Cosine and Jaccard Similarity Measures in Vector Space Model. *International Journal of Computer Applications.* 164(6): 28 – 30.

KANGAS S. 2002. Collaborative Filtering and Recommender Sstems: Research Report. In: *VVT Information Technology*. Espoo: Finland.

KEENAN, J. 2019. Corporate Responsibility and the Social Risk of New Mining Technologies. *Corporate Social Responsibility and Environmental Management.* Wiley Online Library. 26(4): 762 – 760.

KEIM, T. 2007. Extending the Applicability of Recommender Systems: A Multilayer Framework for Matching Human Resources. In Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS). Hawaii: IEEE.

KONSTAN, J. A. AND RIEDL, J. 2012. Recommender systems: from algorithms to user experience. *User Modeling and User-Adapted Interaction.* Berlin: Springer. 22(2): 101–123.

KUMARAN, G. & ALLAN, J. 2004. Text classification and named entities for new event detection. In: *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* Sheffield: ACM.

KUMARAN, G & ALLAN, J. 2005. Using Names and Topics for New Event Detection. In: *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Vancouver: ACL. 121 – 128.

LAUMER, S. & ECKHARDT, A. 2009. Help to Find the Needle in a Haystack: Integrating Recommender Systems in an IT Supported Staff Recruitment System. In: *Proceedings of the Special Interest Group on Management Information System's 47th Annual Conference on Computer Personnel Research*. Limerick, Ireland.

LEVANDOSKI, J. J., SARWAT, M., ELDAWY, A. & MOKBEL, M. F. 2012. Lars: A location-aware recommender system. In: *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*. 450–461.

LIU, J., JIANG, Y., LI, Z., ZHANG, X. & LU, H. 2016. Domain-sensitive recommendation with user-item subgroup analysis. *IEEE Transactions on Knowledge and Data Engineering.* 28**:** 939-950.

LU, J., WU, D., MAO, M., WANG, W., AND ZHANG, G. 2015. Recommender system application developments. *Decision Support Systems.* 74: 12–32.

LU, Y., HELOU, S.E. & GILLET, D. 2013. A Recommender System For Job Seeking And Recruiting Website.In: *Proceedings of 22nd International World Wide Web Conference.* Rio de Janeiro: Companion.

MAJID, A., CHEN, L., CHEN, G., MIRZA, H.T., HUSSAIN, I. & WOODWARD, J. 2013. A Context-Aware Personalized Travel Recommendation System Based on Geotagged Social Media Data Mining. *International Journal of Geographical Information Science.* 27(4): 662–684.

MALINOWSKI, J., KEIM, T. & WIETZEL, T. 2005. Analyzing the Impact of IS Support on Recruitment Processes: An E-recruitment Phase Model. In: *Proceedings of the ninth Pacific Asia conference on information systems (PACIS).* Bangkok: Thailand.

MALINOWSKI, J., KEIM, T., WENDT, O. & WEITZEL, T. 2006. Matching People and Jobs: A Bilateral Recommendation Approach. In: *Proceedings of the 39th Annual Hawaii International Conference on System Sciences.* Hawaii: USA.

MALINOWSKI, J., WEITZEL, T. & KEIM, T. 2008. Decision Support for Team Staffing: An Automated Relational Recommendation Approach. *Decision Support Systems.* 45(3): 429-447.

MARIN, A. & WELLMAN, B. 2011. Social network analysis: An introduction. *The Sage handbook of social network analysis*. Sage Publisher. 11–25.

MONTANER, M., LOPEZ, B. & DE LA ROSA J.L. 2003. A Taxonomy of Recommender Agents on the Internet. *Artificial Intelligence Review*. Kluwer Academic Publisher. 19(1): 285 – 330.

NADEAU, D. & SEKINE, S. 2007. A Survey of Named Entity Recognition and Classification. *Lingvisticae Investigationes*. 30(1): 3–26.

OVIATT, S. & COHEN, P. 2000. Multimodal Interfaces that Process What Comes Naturally. *Communications of the ACM*. 43(3): 45-53.

PAPARRIZOS, I. CAMBAZOGLU, B.B. & GIONIS, A. 2011. Machine Learned Job Recommendation. In: *Proceedings of the Fifth ACM Conference on Recommender Systems.* Chicago. Illinois.

POLATIDIS, N. & GEORGIADIS, C. 2013. Mobile Recommender Systems: An Overview of Technologies and Challenges. In: *Preceedings of 2013 Second International Conference on Informatics & Applications (ICIA).* Lodz: IEEE. 282 – 287.

RICCI, F. 2011. Mobile recommender systems. *International Journal of Information Technology and Tourism.* 12.

RICCI F., LIOR R., & BRACHA S. 2011. *Introduction to Recommender Systems Handbook. Recommender Systems Handbook.* Springer. 1 – 35.

RICCI, F. & NGUYEN, Q. N. 2005. Critique-based mobile recommender systems. *OEGAI Journal.* 24: 1-7.

RIEDL, J.T., HERLOCKER, J.L., KONSTAN, J.A., & TERVEEN, L.G. 2006. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* New York: ACM. 22(1): 5 –53.

SADEH N.M. 2002. *mCommerce: Technologies, Services and Business Models.* New York:Wiley.

SARWAT, M., LEVANDOSKI, J. J., ELDAWY, A. & MOKBEL, M. F. 2013. Lars: A Scalable and Efficient Location-Aware Recommender System. In: Proceedings of the *2013 IEEE 28th International Conference on Data Engineering.* Washington DC: IEEE. 450 – 461.

SEGUELA, J. & SAPORTA, G. 2013. A Hybrid recommender System to Predict Online Job offer Performance. *Revue des Nouvelles Technologies de Information.* France: Hermann.

SEKIGUCHI, T. 2004. Person-Organization Fit and Person-Job Fit in Employee Selection*:* A Review of The Literature. *Osaka Keidai Ronshu*. 54(6): 179-196.

SETTEN, M.V., POKRAEV, S. & KOOLWAAIJ, J. 2004. Context-Aware Recommendations in the Mobile Tourist Application COMPASS. In: *Proceedings of International Conference on Adaptive Hypermedia and Adaptive Application Web-Based Systems*. Berlin: Springer. 3137: 235 – 244.

SHARDANAND, U. 1994. *Social Information Filtering for Music Recommendation.* Massachusetts Institute of Technology.

SHARDANAND U., & MAES P. 1995. Social Information Filtering: Algorithms for Automating Word of Mouth. In: *Proceeding. Conference. Human Factors in Computing Systems.* New York: ACM, 210 – 217.

SIBSON, R. 1973. SLINK: An Optimally Efficient Algorithm for the Single-Link Cluster Method. The Computer Journal. 16(1): 30 – 34.

SINGH, V., TIWARI, N. & GARG, S. 2011. Document Clustering Using K-Means, Heuristic K-Means and Fuzzy C-Means. In: *Proceedings Computational Intelligence and Communication Net- works (CICN) 2011 International Conference*. Gwalior: IEEE, 297 - 301.

SMITH, J. E. & BROWN, A. M. 2005. Building a culture of learning design: Reconsidering the place of online learning in the tertiary curriculum. *Division of Technology, Information and Library Services.* Queensland: Brisbane, 615 – 623.

STAFFORD, T.K., AND GILLENSON, M.L. 2003. Mobile commerce: What it is and what it could be. Communications of the ACM, 46(12).

STEINBACH, M., KARYPIS, G. & KUMAR, V. 2000. A comparison of document clustering techniques. In *KDD Workshop on Text Mining.*

SU, X & KHOSHGOFTAAR, T.M. 2009. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence.* 421-425.

SYMEONIDIS, P., NTEMPOS, D. & MANOLOPOULOS, Y. 2014. Location-based social networks. In *Recommender Systems for Location-based Social Networks.* Springer, 35 – 48.

TANG, K.P., LIN, J., HONG, J.I., SIEWIOREK, D. P. & SADEH, N. 2010. Rethinking Location Sharing: Exploring the Implications of Social-Driven Vs. Purpose-Driven Location Sharing. In: *Proceedings of the 12th ACM international conference on Ubiquitous computing*. New York, USA: ACM, 85–94.

TUAN, K.S. 2019. Intelliget Job Matching with Self-learning Recommender Engine. In: *Proceedings of the 9th International Conference on Applied Human Factors and Ergonomics (AHFE 2019).* Chicago. Illinois. 3: 1956 – 1965.

TUNG, W. & SOO, W. 2004. A Personalized Restaurant Recommender Agent For Mobile E-Service.In: *Proceedings IEEE International Conference on e-Technology, e-Commerce and e-Service*. Taipei, Taiwan: IEEE. 259 – 262.

VILJANAC V. 2012. *Ranking of Facebook Friends Based on User Profiles.* Faculty of Electrical Engineering and Computing. Zagreb.

WANG, C.Y., WU, Y.H. & CHOU, S.C.T. 2010. Toward a Ubiquitous Personalized Daily-Life Activity Recommendation Service with Contextual Information: A Services Science Perspective. *Information Systems and E-Business Management.*8(13).

WANG, H., TERROVITIS, M. & MAMOULIS, N. 2013. Location recommendation in location-based social networks using user check-in data. In: *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. Florida, USA: ACM. 374– 383.

WARNER, J.S. & JOHNSTON, R.G. 2003. GPS Spoofing Countermeasures. *Homeland Security Journal*. 25(2): 19–27.

WERTHNER, H., & KLEIN, S. 1999. Information Technology and Tourism: a Challenging Relationship. *Annals of Tourism Research*. Springer. 29(2).

WIN, T.T. & MON, L. 2010. Document Clustering by Fuzzy C-Mean Algorithm. In: *Proceedings of Advanced Computer Control (ICACC) 2010 2nd International Conference*. Shenyang, China: IEEE. 239 - 242.

WOERNDL, W., SCHUELLER, C. & WOJTECH, R. 2007. A Hybrid Recommender System for Context-aware Recommendations of Mobile Applications. In: *Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering Workshop*. Istanbul, Turkey: IEEE. 871 – 878.

XIAO, X., ZHENG, Y., LUO, Q. & XIE, X. 2010. Finding similar users using category-based location history. *In Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*. New York, USA: ACM: 442 – 445.

XIAO, X., ZHENG, Y., LUO, Q. & XIE, X. 2014. Inferring social ties between users with human location history. *Journal of Ambient Intelligence and Humanized Computing*. 5(1): 3–19.

YANG, W.-S., CHENG, H.-C. & DIA, J.-B. 2008. A location-aware recommender system for mobile shopping environments. *Expert Systems with Applications*. 34(1): 437 – 445.

YE, M., YIN, P. & LEE, W.C. 2010. Location recommendation for location-based social networks. In: *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*. New York, USA: ACM. 458–461.

YE, M., YIN, P., LEE, W.C. & LEE, D.L. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In: *Proceedings of the 34th* *international ACM SIGIR conference on Research and development in Information Retrieval*. New York, USA: ACM: 325–334.

YEUNG, K. F., YANG, Y. & NDZI, D. 2012. A Proactive Personalised Mobile Recommendation System Using Analytic Hierarchy Process and Bayesian Network. *Journal of Internet Services and Applications*. 3: 195-214.

YI, P., YENG, C. & ZHANG, Y. 2016. A Job Recommendation Method Optimized by Position Description and Resume Information. In: *Proceeding of the 2016 IEEE Advanced Information Management, Communucates, Electronic and Automation Control Conference (IMCEC).* Xi'an, China: IEEE. 761 – 764.

YING, J.J.C., LU, E.H.C., LEE, W.C., WENG, T.C. & TSENG, V.S. 2010. Mining User Similarity from Semantic Trajectories. In: *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks.* San Jose, California: ACM: 19–26.

YU, H., LIU, C & ZHANG, F. 2011. Reciprocal Recommendation Algorithm for the Field of Recruitment. *Journal of Information & Computational Science.* 8(16): 4061– 4068.

YUAN, W., SHU, L., CHAO, H.C., GUAN, D., LEE, Y.K. & LEE, S. 2010. ITARS: Trust-Aware Recommender System Using Implicit Trust Networks. In: *IET Communications.* IET. 4(14): 1709-1721.

ZHAO, Y. & KARYPIS, G. 2002. Evaluation of Hierarchical Clustering Algorithms for Document Datasets. In: *Proceedind of the eleventh international conference on Information and knowledge management*. New York, USA: ACM: 515 - 524.

ZHENG, V. W., ZHENG, Y., XIE, X. & YANG, Q. 2012. Towards Mobile Intelligence: Learning From GPS History Data For Collaborative Recommendation. *Artificial Intelligence*. 184: 17–37.

ZUVA, T. & ZUVA, K. 2014. Generic Service Discovery Recommender System for Mobile Users. *International Journal of Future Computer and Communication*. 3(6): 432 – 435.