Android App Store (Google Play) Mining and Analysis

MOHAMMAD TAFIQUR RAHMAN

Stockholm 2013

Software Engineering of Distributed Systems
School of Information and Communication Technology
KTH - Royal Institute of Technology
Android App Store (Google Play) Mining and Analysis

Mohammad Tafiquur Rahman

March 30, 2013
Abstract

The aim of mining and analysis of Apps in Google Play, the largest Android app store, is to provide in-depth insight on the hidden properties of the repository to app developers or app market contributors. This approach can help them to view the current circumstances of the market and make valuable decisions before releasing products. To perform this analysis, all available features (descriptions of the app, app developer information, app version, updating date, category, number of download, app size, user rating, number of participants in rating, price, user reviews and security policies) are collected for the repository and stored in structured profile for each app. This scientific study is mainly divided into two approaches: measuring pair-wise correlations between extracted features and clustering the dataset into number of groups with functionally similar apps. Two distinct datasets are exploited to perform the study, one of which is collected from Google Play (in 2012) and another one from Android Market, the former version of Google Play (in 2011). As soon as experiments and analysis is successfully conducted, significant levels of pair-wise correlations are identified between some features for both datasets, which are further compared to achieve a generalized conclusion. Finally, cluster analysis is done to provide a similarity based recommendation system through probabilistic topic modeling method that can resolve Google Play’s deficiency upon app similarity.
Contents

1 Introduction 4

2 Background Study 6
  2.1 Android ........................................ 6
  2.2 Web Crawling .................................. 7
  2.3 Clustering ..................................... 7
    2.3.1 Supervised Clustering ....................... 8
    2.3.2 Semi-Supervised Clustering ................. 8
    2.3.3 Unsupervised Clustering .................... 9
  2.4 Similarity .................................. 12
    2.4.1 Vector Space Modeling ...................... 12
  2.5 Correlation Analysis .......................... 15
  2.6 Recommender System ........................... 15
    2.6.1 Content-based Recommendation .............. 15
    2.6.2 Collaborative Filtering Recommendation .... 16
    2.6.3 Hybrid Recommendation .................... 16
  2.7 Analysis Toolkits ............................ 16
    2.7.1 Machine learning for language toolkit (MALLET) ... 16
    2.7.2 Clustering Toolkit (CLUTO) .................. 17
  2.8 Summary .................................... 17

3 Problem Description 19

4 Analysis Approach 20
  4.1 Data Extraction ............................... 20
  4.2 Data Parsing .................................. 21
  4.3 Analysis Approach ............................. 21
    4.3.1 Correlation Analysis ......................... 21
    4.3.2 Cluster Analysis ............................ 21

5 Experimental Results 23
  5.1 Dataset ...................................... 23
  5.2 Results ..................................... 25
    5.2.1 Correlation Analysis Results ............... 25
    5.2.2 Cluster Analysis Results ................... 26

6 Recommendation based on App Similarity 30

7 Conclusion and Future Work 35
List of Figures

2.1 A Web Crawling Architecture (standard) .......................... 8
2.2 Graphical representation of clustering [19] .................. 9
2.3 Supervised clustering technique, where filled and empty circles represent two different items [21] .................. 10
2.4 Unsupervised clustering technique, where filled and empty circles represent two different items [21] .................. 11
2.5 The graphical model for latent Dirichlet allocation [32] where \( \alpha \) (dimensionality vector) and \( \beta \) (word probability) are the Dirichlet parameter for word and topic distributions. .................. 12
2.6 Classification of IR techniques [41] .................. 13
2.7 Angle between document A and B in two dimensional space [43] 13
2.8 Graphical demonstration of Lesk algorithm [46] .................. 14
2.9 The semantic diagram of CLUTO clustering toolkit .................. 17
4.1 The overall architecture for App analysis .................. 20
5.1 Quantity of Apps in each category for Small dataset .................. 24
5.2 Quantity of Apps in each category for Large dataset .................. 24
5.3 User participation in app rating for Small dataset .................. 25
5.4 Distribution of user ratings provided over Apps for Small dataset .................. 25
5.5 Representation of negative correlation between Download and Price for Small dataset .................. 28
5.6 Representation of strong positive correlation between Download and Participation for Small dataset .................. 28
5.7 Similarity percentage for apps in few categories for Small dataset .................. 28
5.8 Price and Rating correlation between Large dataset (left) and Small dataset (right) .................. 29
5.9 Performance of clustering algorithm across different cluster size .................. 29
5.10 Distribution of number of app developers in determined clusters .................. 29
6.1 Developer Contribution over App category .................. 30
6.2 The working procedure for similarity based recommendation system .................. 33
6.3 Number of categories in clusters .................. 33
List of Tables

2.1 Sample Topics .................................................. 17

4.1 Example of the topics identified from App descriptions obtained using LDA technique ........................................... 22

6.1 The app in rank 0 is the candidate app (*Learn HTML*), where others are 10 most similar apps for *Learn HTML* according to the Lesk similarity coefficient. .................................................. 32

6.2 Profile for *Learn HTML* ........................................... 34
Chapter 1

Introduction

Smart-phone applications, known as Apps, are the third party software applications that have some specific objectives, requirements and capabilities [1]. They provide the opportunity to use the interface of passive content with the facilities of navigating, searching, rating, commenting on and sharing published contents [2]. Apps are stored in large software repositories, which are formally named as App Stores in online app market. The main three app repositories are Google Play\(^1\), iPhone App Store\(^2\) and BlackBerry App World\(^3\), among which Google Play has been chosen for analysis because it is the most popular and fast growing service provider in app industry. According to Sabatini [3], 72% of the total Android apps (globally), which are available in Google Play, are offered with no cost that helps Google to lead the business.

According to Harman et al. [4], information repositied in app store is categorized into three different areas: app developer and customer perspective, business view and technical points. Customers’ participation is considered in form of rating, participation in rating, user review and tags. The business view contains apps’ statistical and organizational information, for example, category, price, number of download and size of each app with recommendation to suggest some related apps from a large number of choices. Finally, the technical and feature information is presented in the description section in textual format. So it is necessary to employ data mining technique to extract technical information [4] that is necessary to group functionally similar applications into same cluster.

This rich collection of data into a single repository produces an informative and inter-related dataset with the opportunity to analyze and understand the relationship among dataset members (i.e., apps). This analytical result of inter-related data will provide some insights to the application developers or managers to realize the app market and consider changes to features before releasing new products or updating versions and these insights cannot be achieved by crawling only [4].

This study is conducted by adopting and extending the approach of Harman et al. [4] on software repository mining. For analysis, a large corpus of data is constructed, where data is collected from different sources, for example, end users, app developers and the app store itself. The analytical work is done in

\(^1\)https://play.google.com/store
\(^2\)http://www.apple.com/iphone/apps-for-iphone
\(^3\)http://appworld.blackberry.com/webstore/
three continuous steps: extraction from the repository, data parsing for feature extraction and pair-wise correlation analysis. First of all, the available apps are crawled to gather raw data from the repository (i.e., webpage for each app). Second of all, collected raw data is parsed to fetch feature information for app profiles. Last of all, correlation analysis is done between pairs of features to determine how strongly they are related with each other. Other than correlation analysis, a recommendation system is developed to suggest top similar apps to the candidate app. Probabilistic topic modeling method [5] has been utilized to cluster apps into functionally similar groups that helps the recommendation system to suggest technically similar apps.

The following Chapter 2 represents the background study, where generalized information on exploited technologies and techniques are provided. An overall discussion on the problem definition is presented in Chapter 3, where Chapter 4 deals with the approaches that are utilized to retrieve and parse (feature extraction) information and analyze the pair-wise relationship between features. In Chapter 5 experimental results and discussions are exhibited. Chapter 6 demonstrates the method for similarity based recommendation. The overall findings of this analytical work is summarized and ideas on future work are expressed in Chapter 7.
Chapter 2

Background Study

Android supports extensive third-party applications. Current statistics says that the number of Android app is approximately 700,000 [6] and has captured around 75% of the smart phone market share [7]. Google Play provides a rich source of information related to its customers, business and technical features. Among the structured information (for example, rating, price, downloads), it provides the technical information in description part in an unstructured text format [4] that requires data mining technique to extract information for analysis.

2.1 Android

Android is a mobile software program with Linux operating system, middleware and key applications that are developed by using the libraries of Java language [8]. It can access phone hardware (such as, phone camera) and user information stored in the phone (such as, phone book, call history, short message etc.) by employing its API (Application Programming Interface) [9]. These software, which are formally known as Apps, are stored and made available in an online repository that is called App store, where user able to browse, download and rate them and can also leave comment on them [10]. Google Play is the official repository for Android apps, where several other third-parties (e.g., Amazon\(^1\), Appbrain\(^2\) etc.) also exist into the market to make the applications’ APK (application package) files available to the end user to download and install. Its open development platform enriches the repository with hundreds of thousands of applications. Besides making apps available for download, the app store also provides information on average rating, participants in rating, user review, description and features, screenshots, number of download, price, app version information, Android version that is suitable to run a particular app and permissions. Google Play not only delivers the above information but also provides different types of recommendation based on the usage and development of apps that are related to the currently viewing app. The Google Play app store is mainly divided into Game and Application groups, where Application is then sub-divided into 24 categories. Users can also search app through Top Free and

\(^1\)https://www.amazon.com
\(^2\)https://www.appbrain.com
Top Paid category that can make users’ searching much easier and less time consuming.

2.2 Web Crawling

Web crawling is an automated system that is used to download large amount of data from the World Wide Web [11]. This automated system is a computer program or script that searches for the desire data through scanning internet pages. It normally runs only once for data collection but it is not unusual to run the crawler in periodic manner. For example, Google crawls the web regularly to modify or rebuild its index. Its computer program decides on crawling websites, frequency of crawling and number of fetching pages from each site [12]. It was first used by Matthew Gray in 1993 to generate first web search engine written in Perl scripting language [11]. It is still widely used as one of the main components of modern search engines to provide relevant websites and/or information that are available on the public web pages to the users according to their queries. It is also used for web archiving, data mining, web monitoring, linguistics, market analysis and organized manner searching. Information is changing rapidly and large numbers of new web pages are being added every day with various content or existing web pages are being modified with new information. So, for keeping the search engines’ databases updated with newly added information, web crawler is the effective way. But it is also used illegally to gather unauthorized information, for example, server hacking, blogs, restricted sites [13]. Figure 2.2 demonstrates a standard architecture of a web crawling system.

2.3 Clustering

Clustering is an information retrieval technique that puts the items physically together, which are logically similar [14]. In detail, it groups the common characteristics sharing objects together in multi-dimensional space and shows dissimilarity with objects in other clusters with different characteristics at the same time [15]. So, this technique is considered as a tool that can efficiently perform data reduction by creating more manageable subgroups, so that data indexing, filtering, searching, mining and in general, information retrieval becomes easier and faster [16]. The clustering techniques were first used in life science [17] but its application has been expanded to other subject areas, for example, Mathematics, other wings of science, business and especially in World Wide Web. Researchers of different fields of interests, for example, statistics, pattern recognition, data mining, image analysis, bioinformatics and machine learning widely use clustering to perform learning on hidden data concept in a simplified way [18]. Figure 2.3 demonstrates the clustering concept.

Clustering data into different similar groups can be done mainly in three different ways: Supervised, Semi-supervised and Unsupervised clustering. Among these approaches unsupervised clustering is widely and traditionally used to retrieve information from unstructured textual documentations. Each of these approaches is elaborated in the following sections:
2.3.1 Supervised Clustering

Supervised clustering is applied on sets of items that are classified to produce complete clustering over these sets that are similar to a single class [20],[21]. This clustering technique tries to maintain the class purity by generating different clusters for different class labels and keep the total number of clusters low at the same time [21]. Figure 2.3.1 illustrates the supervised clustering technique, where it separately clusters items A (filled circles) and B (empty circles) that differentiates it from the traditional unsupervised clustering.

2.3.2 Semi-Supervised Clustering

Semi-Supervised clustering approach is applicable to datasets, where a small amount of classified knowledge is available that is used to optimize the class purity by guiding the traditional clustering process [22],[21]. This clustering approach is sub-divided into two different groups: similarity-based and search-based, which are differentiated by their information combining techniques [22]. The first method adapts the similarity measure of traditional clustering technique, where the available classified knowledge is incorporated and the last method modifies the clustering algorithm to achieve appropriate clusters [22],[21].
2.3.3 Unsupervised Clustering

Unsupervised clustering is the traditional approach to cluster documents, where classified knowledge on datasets is completely unavailable. It identifies groups of items such that the level of similarity between the items within a group is higher than that of with items into other clusters [22]. Figure 2.3.3 represents the unsupervised clustering technique that maintains a low level of purity by putting items A (filled circles) and item B (empty circles) into same cluster or B into two distinct clusters. Unsupervised clustering is done through hierarchical, partitional and probabilistic ways that are explained in the following.

Hierarchical Clustering

The hierarchical method produces a set of nested clusters that are merged to make a single large cluster or a single large cluster is split into a set of nested clusters. The initial one is known as Agglomerative approach, where a series of N-1 agglomerations of pairs of objects are used and the latter one is called Division approach, where all of the objects in a cluster are taken into account and divided into two smaller clusters at each N-1 steps. This division continues until each object finds its own cluster [14].

Partitional Clustering

The partitional method of clustering is initially associated with the pre-determined number of desire disjoint clusters (M), where the available objects are assigned. In partitional methods the classes are mutually exclusive [14]. The partitioning process continues until a particular objective function is optimized and this optimization causes the creation of high quality clusters [23]. K-means clustering is the commonly used partitional clustering method, where the required number of clusters, which is equal to the number of components in the population, is
Figure 2.3: Supervised clustering technique, where filled and empty circles represents two different items [21].

given. Each component in the population is further examined and assigned to a cluster by considering the minimum distance between components and clusters. As soon as, a new object is added to a cluster the centroid position is recalculated and this process continues until all the objects are associated with the pre-determined number of clusters [24]. Bisecting K-means is another method for partitional clustering, where the collection (that can be considered as a single large cluster) of documents is split into two separate clusters and bisect further by splitting documents into a total of three clusters and so on. This process continues until the desired number of clusters has been achieved [25].

**Probabilistic Clustering**

The probabilistic method of clustering works with datasets that are consistent with the available knowledge in the training set [26], are accumulated from a mixture of several distributions [27]. Each distribution of the mixture is further modeled over latent topics [28] and these latent topics can be generated by employing topic modeling, a widely used data mining technique to make a summary of the collected data.

**Topic Modeling:** According to Griffiths and Stryvers [29], topic models are based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words. Topic models are generative probabilistic models for document, which use vocabulary to identify topics within text corpora. In the recent history, topic modeling technique is used for different purposes, for example, Newman and Block [30] have employed topic models on the collection of newspaper articles for topics and trends discovery that oc-
Figure 2.4: Unsupervised clustering technique, where filled and empty circles represent two different items [21].

...occurred over time, where Griffiths and Steyvers [29] used it for searching the research topic trends by concentrating scientific papers' abstracts and Hall et al. [31] employed it for finding the computational linguistics trends. Topic modeling over documents is usually done by utilizing the Latent Dirichlet Allocation (LDA) technique that was first proposed by Blei et al. and was considered as the first fully probabilistic model for text clustering [32]. It is an evaluation of pLSI (probabilistic Latent Semantic Indexing) with increased modeling flexibility [33] that partly solved the LSI's (Latent Semantic Indexing) lacking over solid probabilistic foundation.

Latent Dirichlet Allocation (LDA): It is an unsupervised, statistical, approach to document modeling that discovers multiple latent semantic topics in large collection of text documents [34]. It uses an algorithm with the aim of finding short description of the collected data by mining several times occurring sets of words from text corpus and these sets of words are used as topics, the building blocks for these short descriptions. LDA employs Bag-of-words modeling technique for categorization, where each document and topic is represented as probability distribution over topics and a number of words respectively [35]. This topic distribution can be inferred for arbitrary document by calculating the maximum-likelihood estimates for model parameters, where the corpus is used as input [36]. The application area of LDA includes information retrieval, image processing [37], web mining, collaborative filtering and spam detection [34]. The web search engines including Google are using this technique in order to perform better determination of user intent by evaluating the search words as a unit. The LDA process is graphically illustrated in figure 2.5, where rectangles represent replicates (larger one is the collection of documents and smaller one is...
for each document from where topics and words are chosen) and circles denote Dirichlet parameters [32]. For each of \((N)\) documents from the collection of \((M)\) documents, the process firstly picks up a vector \((\theta)\) with potentially appearing topics. Secondly, a topic \((z)\) is drawn from the chosen vector for each of the words in that document and finally, a word \((w)\) is drawn from the multinomial probability distribution for the chosen topic [34].

![Graphical Model for Latent Dirichlet Allocation](image)

**Figure 2.5**: The graphical model for latent Dirichlet allocation [32] where \(\alpha\) (dimensionality vector) and \(\beta\) (word probability) are the Dirichlet parameter for word and topic distributions.

### 2.4 Similarity

#### 2.4.1 Vector Space Modeling

It is one of the formal, feature-based, individual, partial match information retrieval techniques, where vectors of weighted term counts are used to represent documents and queries. The closeness between document and the query document is calculated through its integral similarity function [38] that is further used to produce a list of documents relevant to the query document [39] and the similarity is determined by finding the matching (even partial) words in a pair of documents [40]. Among multiple similarity measures Cosine similarity, Dot product similarity and Lesk similarity are studied to serve the purpose. Figure 2.4.1 demonstrates the classification of information retrieval techniques, where Vector Space Modeling (VSM) is placed at detail level under formal model of feature-based category [41]. The studied similarity measures are described as follows:

**Cosine Similarity**

The Cosine similarity measure uses the angle (figure 2.4.1) between query and document for similarity calculation by computing the inner product between them [42]. In this approach, query and document are represented as term vectors in a vector space model, where their correlation is calculated by the similarity measure [43]. Among all other similarity measurement techniques, cosine similarity is considered as the most widely used method for information retrieval and clustering [43]. The cosine similarity between two vectors \((A,B)\) is measured as equation 2.1:

\[
\text{Cosine Similarity} = \frac{A \cdot B}{||A|| \cdot ||B||}
\]
\[ \cos(\theta) = \frac{A \cdot B}{|A||B|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} \] (2.1)

Here, \( A \) and \( B \) are denoting topic models while \( A_i \) and \( B_i \) are referring to words in this topic models.

But this approach faces polysemy (a word with more similar meanings e.g. driving a car and driving results) and synonymy (different words with same meaning e.g. car insurance and auto insurance) problems [44]. The user query terms might not be able to return the perfect match if multiple words represent a single meaning because of word choice differences or might return match with wrong determination if a word has multiple meanings.

**Dot product similarity:**

The Dot product similarity measure computes the Euclidean distance between query and document with a ruler in two or three dimensional space [43]. Ac-

Figure 2.6: Classification of IR techniques [41]

Figure 2.7: Angle between document A and B in two dimensional space [43]
cording to Huang [43], Euclidean distance is standardized to solve geometrical problems but it is also exploited for information clustering. It measures the distance between centroids of text documents that are represented by their term vectors. The distance between document A and B is measured as equation 2.2:

\[
D_E(t_a, t_b) = \sqrt{\sum_{i=1}^{m} (W_{t,a} - W_{t,b})^2}
\]  

where \( (t_a) \) and \( (t_b) \) are the term vectors for document A and B respectively.

**Lesk Similarity:**

The Lesk similarity is measured by identifying the meaning of a word that is used in a sentence, where multiple dictionary meaning for the word is available. Lesk [45] proposed an algorithm to disambiguate word sense by using dictionary meaning in 1986. The intention behind this algorithm is to exploit word sense for identification of words that are not only directly overlapped but also similar in sense. To determine the sense of a word, each of the dictionary meaning of this word is compared with the dictionary meanings of other word in a phrase and a word is selected for the sense that covers maximum number of words with common dictionary meaning to other words in this context [46]. For example, the word *bank* has two senses: financial sense similar to *funds* and river sense similar to *slope*. After examining other related words, the algorithm finalizes the sense that is used in this context. The figure 2.4.1 graphically illustrates the Lesk algorithm. Furthermore, the Lesk similarity measure is not only suitable for semantic networks but also workable with any word definition providing dictionary [47].

![Graphical demonstration of Lesk algorithm](image)

Figure 2.8: Graphical demonstration of Lesk algorithm [46]
2.5 Correlation Analysis

Correlation analysis is a technique that is used to compute and analyze the linear or nonlinear relationship between two continuous variables. This analysis also tries to quantify and interpret the strength of the computed relationship between those variables [48]. There are mainly two methods available to perform the correlation analysis: parametric and nonparametric method.

Parametric method of correlation analysis is a statistical technique that requires some important assumptions, for instance, the elements are normally and uniformly distributed, are to be established [49]. This method is the commonly employed statistical technique for correlation and trend analysis [49] and can be used to summarize a large amount of information if the assumptions are correctly made [50]. In parametric method, Pearson correlation coefficient is used to measure the strength of association between two linearly related variables [48] by utilizing standard deviations.

In contrast, non-parametric method is applied to analyze data, where distribution is not required [51] that relaxes the assumptions [49]. As it is free from distributional assumptions, it can provide more accurate and valuable information [50]. Non-parametric method of correlation analysis works well when datasets are continuous and for this reason, it uses Spearman rank order correlation coefficient to measure the monotonic relationship between two variables [48], where increase in value of one variable causes the value of other variable to increase or decrease.

2.6 Recommender System

Recommendation system is a web application that produces recommendations with a list of products or items to the users related to their choice or interest. For example, Amazon and Netflix are practicing the industry-strength recommendation system for suggesting books and movies respectively [52]. The system suggests those products that are similar to the products user visited, ranked, purchased or downloaded previously, so that the user can receive more information to gain knowledge on their desire products [53]. The recommendation system is mainly classified into three categories, such as Content-based, Collaborative Filtering and Hybrid recommendation approaches based on the way recommendations are made [54]:

2.6.1 Content-based Recommendation

According to Pazzani and Billsus [53], in Content-based recommendation system product description is used to find the similar products to which user might be interested. For this reason, content-based recommendation is related to information retrieval system by filtering out the products or items that are already purchased or visited by the user. Euclidean Distance Metric and Vector Space Model (VSM)’s cosine similarity measure is used to make a list of recommending items, where cosine similarity is the appropriate choice for this recommendation technique because it deals with the description of items (i.e., the description of Android apps). In such cases, neighbor items will be selected by identifying the larger positive cosine value because larger cosine value indicates smaller an-
ingle and therefore smaller distance between items. In this way, the system can recommend similar items related to the currently visiting item to the user. To make good recommendations, the repository should contain enough distinguishing information on items.

2.6.2 Collaborative Filtering Recommendation

The Collaborative Filtering (CF) recommendation system finds out the items that are liked or rated by other users of same taste. CF technique of recommendation is divided into two different approaches: Neighborhood-based approach, where a set of similar users are chosen and predictions are generated for the current user by combining the rating of those selected users and Model-based approach, where users’ previous ratings are used to induce predictive models [52]. This method also uses the same cosine similarity of VSM as content-based method but here the similarity between vectors of user ratings is measured [54].

2.6.3 Hybrid Recommendation

To avoid limitations of content-based and collaborative filtering system some recommendation systems use the hybrid approach: combining the mentioned methods. This combining process is done through different ways, for example, implement those methods separately and then combine their predictions or include some characteristics of first method into second method and vice-versa or generate a model that includes characteristics from both of the methods [54].

2.7 Analysis Toolkits

In this study, Machine learning for language toolkit (MALLET) is used to perform topic modeling, where Clustering Toolkit (CLUTO) is used to perform clustering for the analysis. Additionally, two other software packages: Perl package for Lesk similarity measure and MATLAB package for Spearman rank correlation coefficient are utilized to perform similarity check and correlation analysis respectively.

2.7.1 Machine learning for language toolkit (MALLET)

MALLET is a java-based package that is developed by McCallum in 2002 [55] for Natural Language Processing, topic modeling and information extraction. It takes the whole corpus as input and clusters the documents into different topics that are generated by accumulating distinct words that are similar in nature. It allows users to provide total number of topics to be produced to choose total number of words per topic at the same time. It uses a built-in (default) stop word list but users can provide their own list upon their choices. After completing the learning and training process it generates topics and measures their contributions in documents in html file format for better visualization and in CSV file format for better tagging and subject indexing. Table 2.1 demonstrates two sample topics with their contents.
2.7.2 Clustering Toolkit (CLUTO)

Cluto is a software package that contains stand-alone programs (vcluster and scluster) to perform clustering low and high dimensional datasets. It not only separates the documents of large dataset into different clusters but also analyzes the characteristics of those clusters. It divides documents into clusters with higher similarity among the data within a cluster and with low similarity to other clusters contents. Cluto was developed at Karypis Lab of University of Minnesota Twin Cities and was first exploited for the analysis of life science dataset [56] but nowadays, it successfully supports dataset clustering of different application areas including datasets of science and commerce, biological applications and information retrieval. The figure 2.7.2 illustrates the flow of work of Cluto toolkit.

![Figure 2.7.2: The semantic diagram of CLUTO clustering toolkit.](image)

2.8 Summary

After completing the detail study on smart-phone application it has been identified that the demand for smart-phone application is increasing continuously with new business opportunities in this industry. This increasing demand is inspiring contributors for more investment to capture the market control. They are trying to provide better service with minimum cost and this is the call for the researchers to discover some new ideas for serving new business needs. Some
researches have been conducted on apps’ security and permission in the recent past. Chia et al. [57] have distinguished apps into three platforms such as free apps, apps that have some similarity with the name of popular apps and apps with mature contents. They have researched to find out the group of apps that requires permission for most of the time to use them and have found that popular apps ask for permission more than average apps. Frank et al. [9] have investigated to find the pattern of permission requested by popular apps, where app rating and number of reviews are used for popularity measurement of apps by differentiating them into low-reputation and high-reputation groups. Felt et al. Felt2011 studied Android applications to determine developers’ behavior upon App privilege setting and found the intention of following least privilege setting by the developers. They identified that around one-third of the total App that they examined are over-privileged among which more than 50% request one extra permission where 6% request more than four redundant permissions. Furthermore, Enck et al. [58] have researched on the usage of personal or phone identifier by taking top downloaded apps into consideration, where Rahmati et al. [59] have explored the usage of iPhone in socioeconomic platform and found out how apps are exploited by different socioeconomic status groups. Zhong et al. [60] have tried to classify Android app market into different types of online market (long tail or superstar) and recommended it as a Superstar market with the contribution of popular apps. De and Chiky [61] have studied to provide an open source recommendation system to SoliMobile Project by utilizing the web mining technique over implicit ratings. Moreover, service providers are always performing some analysis on app market to increase their sell [62].

Information retrieval from unstructured data sources (e.g., emails, source codes, documentations) through mining is another field of research [63]. This mining technique is also applied to software repositories for discovering hidden patterns that cannot be indentified with normal view. Harman et al. [4] have considered app store (Blackberry) as a software repository and successfully applied the data mining technique to retrieve features (rating, price and download) from app descriptions that are presented in natural language. This analytical study extends their works by considering all the features available in the Android app store (Google Play) to generate an overall picture of the market to provide insights to the developers. Furthermore, in this scientific work, a similarity based recommendation system has been suggested by identifying the technical dissimilarity among apps within the same category.
Chapter 3

Problem Description

App analysis covers a wide area of research that involves several different criteria. For example, Google needs to perform some research on app market to maximize its profit, the app developers (individuals and/or companies) require some feedback from the market (e.g., end users) to evaluate and continue their works and the end users need to do some research to find out apps that they are looking for. All of these app market participants require some analytical works that will help them to be satisfied on their needs. By considering these three types of participants in the app market, this study has focused on the market from three different points of views by concentrating on app correlation and continue to the end with the focus on app recommendation.

In the study on app correlation, data from end users, developers and business is combined to build a corpus for current situation analysis of the app market. The information on user reaction is collected from average rating that is done by users who have downloaded apps. Business information becomes available in form of downloads and price and technical information is retrieved from the free hand textual description of apps. This approach will help the app development industry and the business to decide on new products and next release by providing insights.

The recommendation system can attract the users very easily by providing suggestions on products and this is one of the key business strategies. Google Play uses three different recommendation systems, two of which are related to the previous users and the rest one is associated with app developers. In this study it is found that almost all of the developers develop app in the same category, so there is a little chance to suggest apps of different types to the consumers and the existing categorization system of Google Play does not categorize apps of same type all the time. There are some apps available in the store, which are categorized together although they do not show any similarity with other apps in its category. So apps are rearranged into different clusters and recommended based on their similarity by exploiting different standard software toolkits.
Chapter 4
Analysis Approach

The Android app repository analysis approach is divided into three phases. The first phase is associated with raw data extraction from the repository to meet the need of having sufficient amount of data in hand to continue the analysis. In the second phase data parsing technique is utilized to fetch the required features from the extracted raw data in the first phase. Finally, in the third phase the correlation analysis between features and the cluster analysis is done. For cluster analysis, data mining toolkit has been used to extract technical information from the textual description of each app. The figure 4.1 represents the all the steps that are utilized in this scientific study.

4.1 Data Extraction

As the first step of the analysis, a web crawler is employed to accumulate raw data from the repository. The crawling application is divided into three sub-processes. The initial process creates a category list by collecting all available categories, where the *Games* category is excluded as game apps are not considered in this analytical study. The second process of the crawling application converts the observed URL patterns for apps in each category into web addresses and creates a list of them, where web addresses are placed categorically together. The final step of the processing is associated with traversing from app category webpage to associated app pages for embodying app information. As
soon as information (raw data) for each app is collected, the system is ready to parse it to extract desire information.

4.2 Data Parsing

As the second step of the analysis, the retrieved raw data, which is in HTML format, are parsed to extract app features and to store them into the app profile. The extracted features are: descriptions of the app, app developer information, app version, updating date, category, number of download, app size, user rating, number of participants in rating, price, user reviews and security policies. This approach of data extraction and parsing can be applied to other app stores (for example, Blackberry app store [1], Apple app store etc.) with little modification in web address formation to identify exact app and to parse for data representation. It is possible to add or deduct information in the app profile according to the necessity.

4.3 Analysis Approach

In the final step, analysis objectives are pursued in two directions: Correlation analysis and Cluster Analysis.

4.3.1 Correlation Analysis

For correlation analysis, different app features are selected to achieve pair-wise correlation coefficients and the considered features are rank, participation (number of users who have rated), number of downloads, price and size. This provides 10 sets of pair-wise correlation across all categories, for example, \( \langle \text{price, rating} \rangle \), \( \langle \text{price, number of downloads} \rangle \) and \( \langle \text{rating, number of downloads} \rangle \) etc. The required statistical data on features are collected from App profiles that are used to determine the correlation strength of two features through Spearman Rank Correlation technique. The correlation result ranges from (-1) to (+1), where (-1) represents perfect negative association of ranks, (+1) denotes the perfect association of ranks and (0) indicates no association between ranks. When this correlation analysis approach is applied to App repositories, it generates an overall picture of the App market in order to inform developers on market’s current situation along with consumers’ desires and attitude by revealing intrinsic properties of the market [4].

4.3.2 Cluster Analysis

For cluster analysis, the clusters of similar applications are taken into account to find out the similarity among the apps that are placed into same category or the tendency of app developers to develop apps for same category. To solve these queries, it is necessary to examine the alliance among clusters’ characteristics and features of interest e.g., App category and developer by constructing clusters with similar apps. The similarity among apps is calculated by utilizing latent topics [5] that are extracted from App description. These latent topics can be achieved from the unstructured textual description of Apps by employing probabilistic topic model that discloses the hidden thematic structure. This
Table 4.1: Example of the topics identified from App descriptions obtained using LDA technique

<table>
<thead>
<tr>
<th>AppName</th>
<th>TopicID</th>
<th>Topic Words</th>
<th>Topic Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery Channel</td>
<td>141</td>
<td>videos app youtube watch download photos enjoy content official easily</td>
<td>0.138</td>
</tr>
<tr>
<td></td>
<td>85</td>
<td>tv watch shows channels channel live media favorite series network</td>
<td>0.138</td>
</tr>
<tr>
<td></td>
<td>41</td>
<td>quotes life famous world knot quote people popular collection tie</td>
<td>0.103</td>
</tr>
<tr>
<td></td>
<td>88</td>
<td>news latest local sports stories breaking articles video coverage entertainment</td>
<td>0.069</td>
</tr>
</tbody>
</table>

technique has been successfully exploited to discover topics and trends from online journals, news, articles and consumer reviews [64],[35].

In order to identify topics models, Latent Dirichlet Allocation (LDA) [32] variation of generative topic modeling technique is utilized. LDA models each application description as a mixture of topics, which are characterized by distributions over words constituting the examined document [65]. The implicit assumption behind LDA is that a document can exhibit multiple topics. The LDA process on document generation is graphically illustrated in figure 2.5 and described in Chapter 2. LDA is applied to the description of each app, which contains textual materials narrating application functionality. The output is a set of topics represented by a collection of words. As an illustrative example, the determined topic words from the description of Discovery Channel app are demonstrated in table 4.1. Accordingly, this app is associated to four topics: 141, 85, 41 and 88 but in different weights: 0.138, 0.138, 0.103 and 0.069 respectively. Each topic in turn is represented by 10 distinct words of similar types.

After finding latent topics that represents each application, applications are grouped into clusters based on the similarity between their topic models. K-means bisecting clustering technique is recruited to generate 280 clusters in total. In this analysis, the objective of clustering function is to optimize (maximize) topic similarity between applications in each clusters through cosine similarity metric.
Chapter 5

Experimental Results

5.1 Dataset

Two different datasets are exploited to perform correlation analysis, one of which is collected in the recent past where the other one is a year old. The recent dataset is pretty small compared to the old one but they are treated in the same way for analysis.

The first dataset contains data on 21,065 apps that are retrieved from 24 different categories of Google Play in November 2012. The size of the collection seems pretty small in comparison with hundreds of thousands of Android apps that are currently available worldwide. This limitation has occurred due to the recently implemented regulations by Google, where they restrict devices to access only those apps, which are available in the current geographical location of devices. For example, a device with USA origin can traverse only those apps that are available in UK, if it is currently located in UK. In the rest of the study this dataset will be referred as Small dataset.

The second dataset is collected from Frank et al.[9], which is quantitatively larger with 450,933 Android apps. The small dataset includes only 4.67% of the total apps accumulated by the second dataset. Frank et al.[9] retrieved the dataset from Android App Market (the older version of Google Play) in October 2011. As Google did not impose its localization policy at that time, they had the repository as a whole that made them able to construct such large dataset. In the rest of the study this dataset will be considered as Large dataset.

The Small dataset includes all the available features and information for apps, where the larger one is available only with limited number of features such as rating, price and participation in rating. The figure 5.1 demonstrates the distribution of apps in each category for the small dataset, where the Personalization category captures maximum number of apps by covering around 6.42% (1,351 items) of the entire dataset and the Libraries and Demo category occupies only 492 apps thus denoted as the smallest group. Some examples from the Personalization category are: Album Art Live Wallpaper, Real Fingerprint Scanner Lock, Rays of Light, Zipper HD Go Launcher EX Locker etc. Additionally, it is also found that Everyone group accumulates maximum number of applications (9,378) and this group restricts each app (which are in this group) from hosting any user generated contents or permitting users to communicate
with each other or querying users for their locations.

On the other hand, the distribution of apps over each category for the Large dataset is demonstrated in the Figure 5.2, where Entertainment category scores the highest with 57,061 apps, where the lowest category is Library and Demo that occupies only 3,356 apps. According to Frank et al. [9], it is identified that average rating based app's reputation is not considerable for its unreliable measure. For this reason, they combine participation in rating with average rating to trace out the actual popularity of apps.

After observing the Small dataset, a very low user trend on product rating has been identified. The figure 'participation in app rating' demonstrates the scenario by finding out the maximum number of apps (13,384) that are rated by a number of users between 1 and 300. It has also been indentified from the study that more than 50% of Android Apps in Google Play are offered with free of cost (56.5% are free and 43.5% are paid Apps). Furthermore, the observation finds out that popular apps are generally smaller in size, which is less than 30,000 kb. Finally, according to the Figure 5.4, the total of 1,813 apps are rated with the average rating of 4.4 out of 5.0, while most of the apps are rated within the range of average rating of 3.8 and 4.8.
5.2 Results

5.2.1 Correlation Analysis Results

Both, Large and Small, datasets are exploited for correlation analysis to compare the trends that will help to assume the findings if it is somehow manageable to increase the total number of app in the Small dataset to the total number of app existing worldwide.

The Small dataset provides no considerable correlation between rating and any of number of downloads, participation and App Size. It follows the same trend for the correlation between size and rating, price, download and participation. These observations suggest that users neither provide rating for the exploited apps nor size sensitive.

The figure 5.5 demonstrates a strong negative correlation between Download and Price with the correlation coefficient of negative 0.6757. The observation also provides same correlation trend between Participation and Price with coefficient of negative 0.4810. This study indicates that if the price of app increases,
the number of downloads decreases, which makes less participation in rating. It is actually pointing out that customers’ concentrate more on free apps than the non-free apps for all categories.

Furthermore, the observation provides a correlation between Download and Participation, where the correlation strength between them is very high with a measurement of positive 0.9 or above for all categories. This scenario is graphically represented in figure 5.6. This strong positive correlation between Download and Participation indicates that the users, who have downloaded and experienced apps, participate in rating.

Additionally, the study also concentrates on apps of each category to measure the degree of similarity among them. The figure 5.7 clearly represents the percentage of inside category similarity for eight categories showing a generalized view of the whole dataset. It denotes that apps in News and Magazines category show the highest similarity among them (by average similarity 44.77%), while apps classified in Lifestyle category are least similar (by average similarity 5.33%). This observation concludes that apps placed in the same category are not similar to each other all the time.

Finally, the Large dataset is considered for correlation analysis and surprisingly, almost similar trend is found for the correlation between Participation and Price and Participation and Rating that is previously presented for the Small dataset. However, the study of comparison between these two datasets identifies a minor difference in the correlation between Price and Rating. This correlation shows a weak positive correlation (+0.0891) for the Small dataset, while it is a weak negative correlation (-0.1351) for the Large dataset. Nevertheless, the graphs for the same correlation in both datasets produce an identical trend that denotes the possibility of producing same correlation analysis result as the large dataset, if the Small dataset is enriched. The figure 5.8 represents the scenario by placing Price and Rating correlation graphs (for both cases) together.

### 5.2.2 Cluster Analysis Results

The Small dataset is divided into numbers of clusters to group apps that are technically similar and topic modeling technique is utilized to produce structured descriptions for apps, which are exploited for similarity check.

The cluster analysis starts by activating the topic modeling process to generate numbers of topics that are constructed by accumulating some structured words. As mentioned previously, MALLET toolkit is employed to perform topic modeling tasks in this study. The toolkit is trained with 20,409 properly constructed app profiles. No certain rule has been identified to decide upon the size of the set (i.e., the total number of topics to be extracted). In this situation, Newman’s [66] guideline for topic quantity estimation is followed, where he suggests to produce 200 topics for 10,000 to 100,000 documents and it is better to construct each topic with 10 distinct words. As soon as the modeling is completed by following Newman’s guideline, each app is represented as an accumulation of few of these 200 topics, where each topic is associated with its weighted contribution to the newly generated app document.

When topics and their contributions are identified for each app, CLUTO clustering toolkit [67] is employed to split the dataset into clusters with similar apps. The toolkit is provided with a mandatory matrix file, where each row is generated with contributed topics and their weights for each app. For instance,
row number one of the matrix file contains topics and their weights for the first document, where row number zero is constructed with total number of rows, total number of columns and total number of non-zero entries respectively. This toolkit also lacks with the guideline for number of clusters to generate and this is the cause to execute the process for several times to produce clusters with different sizes. Finally, to decide on the appropriate number of clusters the internal similarity ($ISim$) and the external similarity ($ESim$) for each cluster is collected to calculate the harmonic average (F-Measure), where $ISim$ denotes the internal similarity among the object of a cluster and $ESim$ indicates the measure of how distinct this cluster is from others. F-Measure is calculated according to the equation 6.1, where $n$ denotes the total number clusters for each time.

$$F - Measure = 2 \cdot \frac{(\sum ISim_n) \cdot (1 - \sum ESim_n)}{(\sum ISim_n) + (1 - \sum ESim_n)}$$ (5.1)

The figure 5.9 summarizes all of the clustering efforts, where cluster size 280 is considered as the best choice for its highest F-Measure score.

As soon as the clusters are identified, their characteristics are further examined from multiple points of views. Firstly, they are inspected for rating, where Phone Calling related cluster comes first with average rating of 4.85 and Music comes after it with average rating of 4.80. Secondly, the cluster related to Providing latest updates for different phones and SMS based comes one after another with the score of average participation of 628.33 and 463.3 respectively. Finally, developers’ contribution is observed by plotting it for each cluster, where it denotes that each cluster is contributed by at least 10 different developers and few clusters are contributed by more than 120 different developers. It has been identified that each cluster is represented by 20 to 40 developers in average that causes developers to deliver best quality app for their survival in this competitive app market. The result is summarized in figure 5.10.
Figure 5.5: Representation of negative correlation between Download and Price for Small dataset.

Figure 5.6: Representation of strong positive correlation between Download and Participation for Small dataset.

Figure 5.7: Similarity percentage for apps in few categories for Small dataset.
Figure 5.8: Price and Rating correlation between Large dataset (left) and Small dataset (right).

Figure 5.9: Performance of clustering algorithm across different cluster size.

Figure 5.10: Distribution of number of app developers in determined clusters
Chapter 6

Recommendation based on App Similarity

Google Play provides three different types of recommendation to its customers, two of them exploit the Collaborative Filtering (CF) technique \[68\] by taking user data (profile) into consideration, where the last one recommend apps from same developer that the user is viewing. CF assumes that users with similar profile show some similarity in their interests upon item selection and this can be done only if sufficient information on users' past experience with apps is available. The figure 6.1 demonstrates the analytical result of developers' contribution for the Small dataset, where around 90% of the total developer contribute for app development in one category only and less than 10% of them work for multiple categories. Additionally, according to the figure 5.7, it has already been traced out that apps in same category are not technically similar. These findings point out that the recommended apps, which come from same developer are influenced by the category of his/her interest while apps for a category are not pretty similar with each other. This scenario causes users not to receive recommendation upon similar apps or to obtain recommendation by utilizing CF technique.

To solve the identified problem a similarity based recommendation system

![Figure 6.1: Developer Contribution over App category.](image)
has been developed that uses a ranked list of apps, which are similar to the candidate app. Initially, App name of user choice is exploited to find the appropriate cluster of similar apps. As soon as the cluster is identified, the app is represented as a set of topic words that is used to measure its similarity with other apps residing in the same cluster by applying Lesk algorithm and all the similarity coefficients are stored in a list. Finally, the list of similarity coefficients is rearranged and ranked to obtain the ranked list of similar app for recommendation.

To analyze the similarity based recommendation system, an arbitrary app Learn HTML is chosen. From the profile for Learn HTML app illustrated in table 6.2, it is identified that Google Play places it in the Book and Reference category and its developer is Thu Gian. The recommender system starts by taking the app Learn HTML as input and generates result with 10 (in this case) most similar apps to the candidate app Learn HTML. Names and particulars of the resulted app are presented in the table 6.1. Firstly, the system generates an app document by accumulating all the topic words that are included in the topics (generated by topic modeling), which are used to represent the candidate app. The app Learn HTML is represented by six different topics (topic 121, 38, 105, 117, 100, and 92), so the app document for Learn HTML is produced by combining all words associated with these six topics. Secondly, the system searches the candidate app in the available clusters (generated by CLUTO) of apps until the app is found out. As soon as the cluster is traced out, the system opts the apps of this cluster for similarity check with the candidate app. In this case, the system traces out that the app Learn HTML resides in the cluster number 222 and chooses 123 other apps available in this cluster for similarity check. Thirdly, the system generates app document for each app in the opted cluster in the similar way it is done previously. This generated app document is compared with the candidate app document and the calculated similarity coefficient is stored into an array. So, the app document for Learn HTML is evaluated with the 123 other apps and the weight for each evaluation (i.e., the similarity coefficient) is stored for further processing. Finally, the repository of similarity coefficient is sorted in ascending order and the associated apps are ranked for recommendation. In this way, apps in the table 6.1 are selected for recommendation to the app Learn HTML. The whole procedure of this similarity based recommendation system is represented in the figure 6.2.

After analyzing the characteristics of the cluster number 222, it is identified that the topic 121 (mostly related with web browsing) is commonly used to represent all of the apps with different contributions. In order to calculate the similarity coefficient between Learn HTML and other apps in this cluster, Lesk similarity measure is exploited, where relations between word senses are considered. Which means this algorithm does not count the words that are directly overlapped, rather count the number of words that are covered by dictionary definitions of words. For this reason, despite of having only one topic in common Dolphin Webzine is selected as the most similar app to Learn HTML, where the fifth similar app CSS Pro Quick Guide has three topics in common.

Furthermore, the cluster is accumulated with the apps from different categories. For example, app Learn HTML, Dolphin Webzine and CSS Pro Quick Guide are from Book and Reference category, where app Dolphin Tab History is from Communication category and Webmaster’s HTML Editor is from Productivity category. This situation cannot be achieved with the current recom-
Table 6.1: The app in rank 0 is the candidate app (*Learn HTML*), where others are 10 most similar apps for *Learn HTML* according to the Lesk similarity coefficient.

<table>
<thead>
<tr>
<th>Rank</th>
<th>AppName</th>
<th>Category</th>
<th>Topics used</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Learn HTML</td>
<td>Book and Reference</td>
<td>121, 38, 105, 117, 100, 92</td>
</tr>
<tr>
<td>1</td>
<td>Dolphin Webzine</td>
<td>Book and Reference</td>
<td>121</td>
</tr>
<tr>
<td>2</td>
<td>Dolphin Tab History</td>
<td>Communication</td>
<td>121</td>
</tr>
<tr>
<td>3</td>
<td>Dolphin Password Manager Lite</td>
<td>Communication</td>
<td>121</td>
</tr>
<tr>
<td>4</td>
<td>Dolphin Browser</td>
<td>Communication</td>
<td>121</td>
</tr>
<tr>
<td>5</td>
<td>CSS Pro Quick Guide</td>
<td>Book and Reference</td>
<td>116, 146, 121, 38, 54, 100</td>
</tr>
<tr>
<td>6</td>
<td>Web to PDF</td>
<td>Communication</td>
<td>121, 100, 70</td>
</tr>
<tr>
<td>7</td>
<td>PDF Viewer for Dolphin</td>
<td>Communication</td>
<td>121, 100, 70</td>
</tr>
<tr>
<td>8</td>
<td>Dolphin Translate</td>
<td>Communication</td>
<td>121, 195</td>
</tr>
<tr>
<td>9</td>
<td>Webmaster’s HTML Editor</td>
<td>Productivity</td>
<td>121, 168, 100, 155</td>
</tr>
<tr>
<td>10</td>
<td>Silver Editor</td>
<td>Productivity</td>
<td>121, 168, 100, 155</td>
</tr>
</tbody>
</table>

Recommendation system of Google Play, unless the app developer develops apps for different categories and according to figure 6.1, it is not possible in most of the cases as developers mostly work for the same category. So, this approach of similarity based recommendation system will facilitate the user to choose app from different categories, which is clearly demonstrated in figure 6.3. It has been identified that four clusters contain Apps from almost all categories (21 to 24 categories), where Apps from 11 to 20 categories reside into 102 clusters. About 77 and 65 clusters accommodate Apps from four to six and seven to ten categories respectively, where only 33 clusters captures Apps from one to three categories.

To evaluate the recommendation system several precision checks have been performed for multiple clusters with different Apps. One of those precision check scenarios is illustrated in Equation 6.1, where cluster number 27 is chosen that contains seven Apps with App ID 805, 806, 5084, 9310, 10823, 10856 and 17246, from which App ID 806 is selected as viewing App to which similar Apps will be recommended. User preferred app is selected and its residing cluster identified to rank the content of the cluster based on the similarity coefficient.

Expected (E): 10823, 805, 17246, 10856, 5084
Achieved (A): 805, 10823, 17246, 10856, 5084

\[
Precision(P) = \frac{|E \cap A|}{|E| \cup |A|} \times 100 = \frac{3}{5} \times 100 = 60\% \quad (6.1)
\]

which indicates 60% accuracy of the presented approach.

32
Figure 6.2: The working procedure for similarity based recommendation system.

Figure 6.3: Number of categories in clusters.
Table 6.2: Profile for Learn HTML

<table>
<thead>
<tr>
<th>(Title) Learn HTML (/Title)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(About) About App (/About)</td>
</tr>
<tr>
<td>(Developer) Thu Gian (/Developer)</td>
</tr>
<tr>
<td>(Rating) 4.5 (/Rating)</td>
</tr>
<tr>
<td>(Participation) 32 (/Participation)</td>
</tr>
<tr>
<td>(Update) September 8, 2012 (/Update)</td>
</tr>
<tr>
<td>(AppVersion) 1.3 (/AppVersion)</td>
</tr>
<tr>
<td>(AndroidVersion) 2.1 and up (/AndroidVersion)</td>
</tr>
<tr>
<td>(Category) Books &amp; Reference (/Category)</td>
</tr>
<tr>
<td>(Download) 1,000 - 5,000 (/Download)</td>
</tr>
<tr>
<td>(Size) 780k (/Size)</td>
</tr>
<tr>
<td>(Price) Free (/Price)</td>
</tr>
<tr>
<td>(ContentRating) Everyone (/ContentRating)</td>
</tr>
<tr>
<td>(Description)</td>
</tr>
<tr>
<td>(UserReview)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>(Permission)</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

34
Chapter 7

Conclusion and Future Work

This scientific study has exploited the Android app repository for app analysis and recommendation. For app analysis, all the available features have been considered to identify the pair-wise correlations and found out that app users are not active enough to rate an app after experiencing and are not sensitive upon the size of app. It has been identified that the feature Price encompasses a strong negative correlation with the feature Downloads and Participation, which indicates the price sensitive characteristic of users i.e., if the App price increases, the number of download decreases and causing less participation in app rating. Finally, the analysis has found out a strong positive correlation between Downloads and Participation, which informs that users’ participation in rating takes place after downloading apps. Some findings of the Small dataset have been compared with those of the Large dataset to clarify the strength of the analysis and has observed similar trends in both datasets logically and graphically.

Furthermore, the analysis has identified the deficiency of similarity based recommendation in the categorization system of Google Play and also found out that apps in categories are not always similar. To resolve these issues, this scientific study has analyzed app descriptions to identify the similarity among apps and developed a system to recommend apps by clustering them into groups, where apps in each group possess some identical characteristics. Probabilistic topic modeling technique and bisecting K-means clustering method has been employed to perform app description analysis and clustering respectively. This recommendation system suggests technically similar apps to the users from different categories.

This analysis has been conducted based on the Small dataset with 21,065 apps that are locally available, where hundreds of thousands of apps exist worldwide. For this reason, the result produced in this analysis is only able to help the local developers and market analysts, not as a whole. So for the future work, this dataset will be enlarged by accumulating globally available apps for greater precision achievement, which can then produce accurate insights about the global app market. Nevertheless, this analytical study is not limited to the Android app market only. It can be applied to other app repositories to perform
different analysis by adding other features (such as user reviews) with existing ones to carry out some new findings.
Bibliography


