Leveraging Mobile App Classification and User Context Information for Improving Recommendation Systems

Mingshan Han

Master of Science in Management
(Information Systems)

Management Science

Submitted in partial fulfillment of the requirements for the degree of
Master of Science in Management (Information Systems)

Faculty of Goodman School of Business, Brock University
St. Catharines, Ontario

Abstract

Mobile apps play a significant role in current online environments where there is an overwhelming supply of information. Although mobile apps are part of our daily routine, searching and finding mobile apps is becoming a nontrivial task due to the current volume, velocity and variety of information. Therefore, app recommender systems provide users' desired apps based on their preferences. However, current recommender systems and their underlying techniques are limited in effectively leveraging app classification schemes and context information. In this thesis, I attempt to address this gap by proposing a text analytics framework for mobile app recommendation by leveraging an app classification scheme that incorporates the needs of users as well as the complexity of the user-item-context information in mobile app usage pattern. In this recommendation framework, I adopt and empirically test an app classification scheme based on textual information about mobile apps using data from Google Play store. In addition, I demonstrate how context information such as user social media status can be matched with app classification categories using tree-based and rule-based prediction algorithms. Methodology wise, my research attempts to show the feasibility of textual data analysis in profiling apps based on app descriptions and other structured attributes, as well as explore mechanisms for matching user preferences and context information with app usage categories. Practically, the proposed text analytics framework can allow app developers reach a wider usage base through better understanding of user motivation and context information.

Acknowledgements

I would like to acknowledge the following people who supported me during the time that I worked on this thesis. This research would not have been accomplished successfully without those people.

First of all, I would like to express deep gratitude to my supervisor Dr. Anteneh Ayanso for his patient guidance, enthusiasm, and encouragement. He not only guides me on my research directions but also helps me understand what research is and how to become an independent researcher.

I would like to also thank my committee members, Dr. Danny Cho and Dr. Eugene Kaciak. They gave me insightful comments and pointed out a broader direction and conceptualization for my research. I thank Dr. Nigussie Mengesha who continuously hire me as Teaching Assistant during my stay in the program so I would not be distracted due to financial problem.

Finally, I would like to thank my parents and my boyfriend who always encourage me and comfort me during stressful times of my program. With their spiritual support, I have been to motivate myself to finish this thesis.

Table of Contents

Table of Tables

Table of Figures

Chapter 1 Introduction

1.1 Background and Motivation

Mobile apps play a significant role in current online application environments. eMarketer (2017) reports that the average time US adults over age 18 spend on mobile apps per day is two hours and 11 minutes whereas the average time spent on mobile webpages is 26 minutes. As the statistics show, in-app usage is dominant compared with webpage usage, and the average time spent on apps is expected to increase in the future. Users not only spend long time using mobile apps, but also download apps more frequently. According to a report by Statista (2017a), the cumulative number of app downloads from the Apple app store was 140 billion as of September 2016. For some of the apps, users need to purchase before the download procedure, and for some other apps, users download them for free but need to purchase when they use extra in-app services (e.g., game currency or magazine subscriptions). Therefore, app downloads generate app revenues and benefit marketers. Statista (2017c) states that apps generated a total revenue of USD88.3 billion in 2016. According to the statistics highlighted above, apps have a long daily usage time, a large number of downloads and generate a large amount of revenue, indicating their increasing significance in our lifestyles and as a subject of research in many fields, including Information Systems and Marketing in the Business domain.

As apps can be easily released by developers on app stores and installed by smartphone users, information overload is becoming a major problem. Statista (2017a) reported that the total number of available apps in Google Play app store and Apple app store were 2.8 million and 2.2 million, respectively as of March 2017. Moreover, Statista (2017b) reported that the number of apps in Google Play experienced a dramatic increase from 2009 to 2017 and the total number of apps reached 3.3 million by September 2017. In addition to the large volume, the purposes of app usage are growing and cover nearly every aspect of our lives, such as navigation, social network, entertainment, among others.

Although mobile apps are part of our daily routine, searching for and finding relevant apps is becoming a nontrivial task due to the number and variety of apps. Traditionally, mobile apps are searched by categories, by keywords, or by a ranking list (Cao & Lin 2017). However, the effectiveness of those traditional methods are limited due to several reasons. Searching by keywords cannot activate potential user needs and has a low possibility of serendipity. Searching by categories is not effective because existing app classification schemes of app stores are theme-based, but not purpose-driven in a way that is aligned with user needs (Al-Subaihin et al., 2016). Mobile app ranking lists and ratings lack reliability because app developers may manipulate the rankings by posting fraudulent ratings or sales (Zhu, Xiong, Ge, & Chen, 2013). Moreover, the small screen of mobile devices limits the number of apps that can be displayed despite the large number of available apps in many stores today. Most importantly, these basic methods are not designed to suit all personal traits, usage patterns, task orientation and other context information. Therefore, it is necessary to develop app recommender systems or app search engines with the capability to provide desirable apps to users based on their personal needs and other context information. An effective recommender system can benefit both users and app developers. On the one hand, it benefits users because they do not need to explicitly search by themselves, which saves both time and effort to find apps that most suit their preferences from a large pool of apps. Also, recommender systems can activate unconscious needs and provide serendipity for the users (Girardello & Michahelles, 2010a). On the other hand, app developers or marketers can earn more profit because an effective recommender system increases the possibilities that apps are viewed or downloaded by users, especially for the apps that are not included in the top list.

With the recent advancements in information technologies and an ever-increasing smartphone user-base, a large amount of implicit and explicit data can be leveraged to create an effective app recommender system. This includes user rating, usage pattern data, customer review, app description, and context information. Studies have proved that context information has significant impact on user behavior in terms of mobile device usage (Barnard, Yi, Jacko, & Sears, 2007). Many studies that engage in effectively integrating context information into other types of information have been conducted to further enhance the performance of state-of-the-art app recommendation systems (Davidsson & Moritz, 2011; Karatzoglou, Baltrunas, Church, & Böhmer, 2012; Woerndl, Schueller, & Wojtech, 2007). However, as discussed in (Cao & Lin 2017), the existing research is limited in their approaches to incorporating context information. The common method is to solve a user-app-context multidimensional model where each type of context information is computed as a new dimension (Adomavicius, Sankaranarayanan, Sen, & Tuzhilin, 2005). Since substantial data collection is required in order to alleviate data sparsity issue and various types of context information are preferred to be included, this approach can be computationally prohibitive and not scalable. In addition, as the dimensions and data size are expanded, interpretation of a complex model can be difficult. As a result, context information like location and time are frequently extracted and leveraged in recommendation model while other types of information are rarely considered.

Besides context information, user needs also play a significant role in designing a system (Oulasvirta, 2005). However, current app recommender systems are designed with limited understanding of user needs and contexts, which is a major gap that needs to be addressed in the literature. In addition, all the studies aim at recommending individual app products which may be too specific to express user preferences or behavior. Since a good app classification scheme can facilitate app recommendation (Liu, Song, Baldi, & Tan, 2016), it is easier to target a desired app category instead of a specific app from millions of apps in app stores. In summary, a novel app recommender system is needed for addressing those problems.

In this thesis, I propose a text analytics framework for mobile app recommendation by leveraging an app classification scheme that incorporates the needs of users as well as the complexity of the user-item-context information in mobile app usage pattern. From a design perspective, recommender systems can leverage novel classification schemes of mobile apps in order to align app categories with usage purpose indicating user preferences or context information. Therefore, I propose to enhance app recommendation by incorporating an app classification framework that addresses user needs or usage behavior as well as context information. As discussed before, the current approaches of app classifications in app stores like Google Play are theme-based (Al-Subaihin et al., 2016) without the consideration of two essential features, namely, user needs and context information. In order to fill in this gap, I adopt and empirically examine an app classification scheme based on the hidden motivation of users, labelled as ‘seven shades of mobile’ (Macki & Draper, 2012). I employ text analytics to profile apps into this app classification scheme based on textual app descriptions. Cao and Lin (2017) indicate that collecting and interpreting high quality semantic context information by text analysis could be a superior way, but is currently a challenging problem. Following this research direction, I demonstrate the feasibility of textual analysis for predicting user preferences based on context information in the form of social media status. Overall, my proposed approach categorizes the app recommendation process into two stages. In the first stage, app categories that suit a user’s preferences are selected. In the second stage, apps belonging to that category are further filtered and added to a recommendation list. In addition, a user desired app can be also recommended automatically. The proposed setting is scalable because preferences can be expressed with an understanding of app categories and other context information can be inferred from social media status.

1.2 Research Goal and Contributions

My overall research goal is to improve the effectiveness of current mobile app recommender systems by leveraging a novel app classification scheme that addresses the needs of users as well as the complexity of the user-item-context information in mobile app usage pattern. In this recommendation framework, I adopt an app classification model using text analytics and explore mechanisms for incorporating user context information such as social media status in the app classification scheme. My research falls under the design science research paradigm and the design artifact of methods (Hevner, March, Park, & Ram, 2004). I propose a conceptual framework of app recommender system including the process of completing the recommendation task. The expected theoretical and practical contributions include: 1) integrating an app classification scheme that addresses the various needs of users in online environments. Integrating unstructured data in online recommendation systems is becoming increasingly important in today's big data environment. While prior research has attempted to address this through mathematical models that exploit the item-user-context multidimensional information structure, this approach will not be scalable as the volume, velocity and variety of unstructured data increase. My research attempts to show the feasibility of textual data analysis in profiling apps based on app descriptions as well as other structured attributes; 2) exploring different mechanisms to match app usage categories with user preferences as well as context information to improve the effectiveness of existing app recommender systems; 3) allowing app developers reach a wider usage base and increase the popularity of apps through better understanding of user motivation and contextual information.

The rest of this thesis is organized as follows. Chapter 2 reviews the literature in recommender system and app classification. Chapter 3 presents theoretical foundations and the conceptual framework for the app classification scheme. Chapter 4 explains the data and methodology used which includes unsupervised learning (exploration), supervised learning (prediction techniques) and empirical validation procedures. Chapter 5 presents the discussion of results and implications of this study. Finally, Chapter 6 provides the study's conclusions, limitations, and future research directions.

Chapter 2 Literature Review

2.1 Product Recommendation Systems: Conventional Techniques

Adomavicius and Tuzhilin (2005) provides an extensive survey of the state-of-the-art recommender system approaches and discuss limitations and future extensions. The authors classified recommender systems into three categories, namely, content-based, collaborative and hybrid. In each category, the underlying recommendation techniques are divided into model-based and heuristic-based. Model-based techniques involve training a data set first and conducting predictions on new data set based on patterns detected from historical data. Techniques include Bayesian classifiers, Decision Trees, Neural Network, Logistic Regression, among others. Unlike model-based techniques, heuristics apply simple rules or common sense. For instance, Pearson Correlation evaluating the strength of linear association is frequently utilized to compute the similarities between variables. Though heuristics are not guaranteed to be perfect, they may be more practical and applicable to different application settings and marketing goals.

2.1.1 Content-based Recommendation System

Since content-based recommendation is originated from information retrieval, this approach aims to look for pieces of information in demand from a collection of information resources (e.g., searching information system related books from the entire database of digital library). The main task of information retrieval is to find out information that is most similar to search objectives or queries. Similarly, content-based recommendation aims at retrieving the items that are similar to user interests. Before computing the similarity, features are extracted from items in order to describe contents of both items and user interests. Conventionally, terms and their frequencies are extracted from textual items like webpages (Pazzani & Billsus 1997), documents and news (Lang 1995). Firstly, user preferences are represented by a user profile containing features of items that are highly rated by users or implicitly evaluated to be preferred by users. Assuming a user is interested in the topic of information systems and highly evaluate relevant articles, her profile will contain words that relate to information systems topics, such as 'software', 'programming' and 'internet'. Secondly, all items that have similar features with user profiles are recommended. In this example, documents with frequent occurrences of information systems related words will be recommended. For computing the similarity between items and user profiles, heuristic similarity formula can be used (Lang 1995). Furthermore, a model can be trained to determine whether an item has a high possibility to be similar with a user profile. Pazzani and Billsus (1997) compare naïve Bayesian classifier with other four techniques to help users differentiate an interesting webpages from an uninteresting one given a topic.

Since content-based methods only consider the similarity between contents of an item and user preferences without the considerations of any other user, pure content-based methods have several limitations despite the multiple and advanced techniques that have been used (Shardanand & Maes, 1995). Due to the connection with information retrieval, content-based methods are mainly employed to recommend items embodying textual information. They cannot be leveraged to recommend items when textual features cannot be extracted (e.g., images and videos). It becomes a problem when the system only recommends items with similar contents unless the user changes usage pattern or explicitly updates preferences. In the case of mobile apps, user may not desire apps that have the same functions with the one she already installed. For instance, music lovers may need only one app to listen to music. It is important to stimulate new fields of interests and provide serendipity. In addition, the popularity and quality of items are not differentiated as long as they share the same contents.

2.1.2 Collaborative Recommendation System

In contrast to content-based recommendation that involves computing the similarity between items and user preferences, collaborative recommendation deals with computing similarity between users. An item is recommended to a user when it is liked by other similar users (i.e., users with similar tastes). In heuristic approaches, two users are considered to be similar when they give similar rating scores to items, and the similarities between rating scores can be measured by traditional Pearson correlation (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994), mean squared difference or other formulas originated from the above two methods (Shardanand & Maes, 1995). After similar users are specified, unknown ratings of a user can be predicted by known ratings rated by similar users. For example, user *A* is considered to give high rating for item *A* when user *A*'s similar users have given item *A* high rating.

However, as mentioned in (Billsus & Pazzani, 1998), heuristic approaches are limited in many aspects. When two users do not mutually rate many items, the similarity between two users cannot be computed due to the data sparsity. This raises the issue of reliability because users with similar tastes cannot be identified if they do not rate the same items. Moreover, dissimilar rating patterns can also be informative but are not usually leveraged in heuristic approaches. Consequently, model-based approaches are developed to solve those problems. A collaborative recommendation task can be converted to a prediction model estimating the ratings of items based on existing rating information given by users. A variety of techniques have been applied to implement the prediction model, such as neural network (Billsus & Pazzani, 1998), clustering and

Bayesian network (Breese, Heckerman, & Kadie, 1998). Although model-based approaches have some advantages over heuristic approaches, there remains other issues. For instance, it is hardly possible to recommend any items to new users because the tastes of new users cannot be computed without rating information. Similarly, newly released items will not be recommended until they are rated by some users. Those two issues are labelled cold start problem (Lin, Sugiyama, Kan, & Chua, 2013).

2.1.3 Hybrid Recommendation System

Due to the limitation of pure content-based and collaborative methods, hybrid recommendation methods are becoming common. Burke (2002) define that hybrid recommender systems combine more than one recommendation method to enhance the effectiveness of the system by addressing the limitation of a single method. Burke (2002) summarized seven methods of combinations. **Weighted** hybrid calculates a weighted score of all the results of individual techniques (Pazzani, 1999). **Switching** hybrid prepares several techniques and switch to one based on the situation. **Mixed** hybrid recommends results from multiple techniques simultaneously. **Feature combination** hybrid mainly does experiment on one method while using data set with features extracted from multiple other methods. For example, collaborative filtering normally input information of user ratings but feature combination hybrid supplement information of item features that are normally collected for content-based methods (Basu, Hirsh, & Cohen, 1998). In contrast, user ratings can be incorporated into content-based text filtering like latent semantic indexing (Soboroff & Nicholas, 1999). **Cascade** hybrid first employs one technique to gain a recommendation list and then employs other techniques to further filter the selection. **Feature augmentation** hybrid acquires partial input features from the result of one technique and then applies another technique to proceed recommendation. **Meta-level** hybrid is similar with feature augmentation, only that the entire learned model would become input features of the second technique. Content based user profiles are consistently incorporated with collaborative filtering (Balabanović & Shoham, 1997; Pazzani, 1999). By this way, users who do not mutually rate the same item could still be computed as similar users as long as they have similar tastes since their user profiles are similar. Therefore, the problem of data sparsity is solved to some extent. When a user is not similar to anyone else, this hybrid method can take advantage of content-based and recommend items that accord with user profile directly.

Besides information that are required in content-based and collaborative methods, other types of information facilitating recommendation are also incorporated into the hybrid methods summarized above. Knowledge-based recommendations input item-related or user related knowledge into conventional recommendation techniques (Burke, 1999). Utility-based recommendations incorporate utility function that is explicitly evaluated by users. Demographic information are also used as an indicator to identify a user's preference (Pazzani, 1999).

Hybrid methods have to be selected properly under different situations. The first three methods are implemented independently, while the other four methods are likely to produce a unifying model. Research works mentioned above prove that hybrid methods perform more effectively compared with the pure methods.

2.2 Mobile App Recommendation System

Recommending mobile apps is different from recommending conventional items such as movie, book, article, etc. Conventional recommendation methods discussed in Section 2.1 cannot be employed efficiently due to the following reasons:

Items like movie, book, article are all for one-time usage. For instance, people decide to see a movie in cinema for a short time entertainment. However, mobile apps support customers in their daily routines. Apps installed on smartphone are required to satisfy users' needs on a daily basis for a long period of time or be capable of temporarily responding some special needs. User preferences can be simply represented by features of items in conventional recommendation whereas the preferences of mobile apps vary frequently because user needs constantly change. Furthermore, collecting datasets that reflect user preferences is a challenging task for mobile apps. The cost for app developers to release apps is low compared with publishing items like movies or books. Therefore, as mentioned in chapter one, a large number of apps have already been released and are constantly updated in a rapid speed. However, only a few apps are used by the majority of users whereas substantial apps are used by a small percentage of users, which result in a sparse dataset and a high kurtosis of app usage distribution (Shi & Ali, 2012). As for users who actually installed and used apps, only an extremely small percentage of users explicitly provide ratings (Girardello & Michahelles, 2010b). One of the example is mobile app Evernote that have not been rated by any user for two months since it was released on App Store in May 2012 (Lin et al., 2013). Although user ratings are frequently leveraged as a resource for computing user preferences in both content-based and collaborative methods, ratings are not a reliable source for mobile apps. Advanced data manipulation techniques are required to solve the problem of data sparsity, and datasets besides user ratings are required to be included in the analysis. Overall, the process of mobile app recommendation is complicated due to the complexity of app usage environment and decision making.

Since conventional content-based and collaborative methods cannot be efficiently applied to app recommendation, advanced methods are developed by adding extensions to those conventional techniques. Besides user ratings, customer reviews and app usage patterns indicating customer behaviors and app descriptions are commonly used to facilitate app recommendation. Shi and Ali (2012) substitute days of app usage during a predefined observation time for user ratings in order to conduct model-based and heuristic based collaborative filtering. Kaji, Yano, and Kawaguchi (2011) collected apps review texts for training a recommendation model with considerations of users' context.

As an extension of collaborative, social-based recommendation are developed to recommend apps that are also used by family, friends or any other people socially connected with users. The core concept of collaborative is to recommend items that are also preferred by like-minded users but it is difficult to define and discover users with similar tastes due to the complex environment of mobile app usage. Besides users with similar tastes, users with social connections are also likely to prefer the same apps. A strong relationship between social networks and app installations are testified in empirical experiments (Pan, Aharony, & Pentland, 2011). Recommender systems are designed to enable users to share app-related information on social network service and search apps that are used by nearby people (Girardello & Michahelles, 2010a). Also, as a new approach of developing effective hybrid app recommender system, social-related information are incorporated into content-based or collaborative methods (Costa-Montenegro, Barragáns-Martínez, & Rey-López, 2012).

Another extension is the complexity of techniques. Advanced mathematical model, text mining and machine learning techniques like LDA (Lin et al., 2013) and tensor factorization (Karatzoglou et al., 2012) are frequently employed in this field. The hybrid methods in Section 2.1.3 are extended to sophisticated unifying models with input of various information simultaneously.

Particularly, the major difference from conventional recommendation is the important role that context information plays. The method of incorporating context information into hybrid recommendation system is a challenging task. Studies related to context-aware recommendation are summarized in the following subsection.

2.3 Context-aware Recommendation System

Barnard et al. (2007) proved that context information has a significant impact on user behavior while using mobile devices. The author conducted an experiment to compare the user behavior under different conditions of motion and light. The results showed that two context factors changed the performance of users on two tasks of reading comprehension and word search. As for mobile apps, which are installed on mobile devices, context information also plays an important role. Various types of context information like gender, age and location may have an effect on user behavior. Men tend to use apps related to game, fitness, electronic devices and vehicles while women are interested in shopping, spa and fashion. Elder people are interested in health while kids should use education apps. Different locations will cause different app usages since some of the apps are location-designed (e.g., New York travel guide and Washington driving test in google play app store). Also, conventional recommendation methods cannot capture contextual information because they are dynamic and affect user preferences at any time (Chen, 2005). For instance, users desire different apps when they are at home and at work. It has been proved that context information can affect the performance of app recommender system (Karatzoglou et al., 2012). Moreover, context-aware recommendation solves the new user problem, one of the essential limitations of collaborative recommendation, indicating new users will not be recommended any items until they explicitly rate or actually start using items (Davidsson & Moritz, 2011). Unlike rating information, context information can be collected at the time of starting the recommender system. Overall, we cannot ignore the importance of context information while developing an app recommender system.

Context information has been defined and categorized in many studies. Chen and Kotz (2000, p.3) referred context as "… the set of environmental states and settings that either determines an application's behavior or in which an application event occurs and is interesting to the user" and defined active context as the first type of context that have an effect on application usage, while passive context as the second type of context that may be related to application but less important. Dey, Abowd and Salber (2001, p.10) defined context as "… any information that can be used to characterize the situation of entities (i.e., whether a person, place, or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves" and categorized context information into identity, location and status (or activity). Zimmermann, Lorenz and Oppermann (2007) categorized context information into individuality, time, location, activity and relations. A system is defined as context-aware "if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task" (Dey, 2001, p.4).

I synthesize the definitions in the literature by emphasizing that context information are environmental settings or user-related situations that potentially affect users' mobile app usage. I consider that knowledge-based, utility-based, demographic based recommendations that were mentioned in Section 2.1.3 all belong to context-aware recommendation. I categorize context information into user-related information, mobile device-related information and external environment. User related information indicate the situation of a user, such as demographic, emotional state, activity, location and social surroundings. Software environment is a representative type of device-related information. External environment stands for the objective factors that cannot be controlled by users, such as time, season and weather. Nowadays, context information can be collected through many techniques like sensors and build-in application of the device. For instance, location can be tracked by Global Positioning System (GPS), time can be obtained by built-in clock and sensors can be designed for sensing low-level types of physical context (G. Chen & Kotz, 2000). Thus, it is technically feasible to obtain context information for the development of context-aware recommender system.

While many studies implemented context-aware app recommendation systems, these systems are limited in the variety of context information and the methodology of incorporating context information. As mentioned in 2.1.2, collaborative recommendation compute the similarities between users based on their ratings and apps are recommended to users when they are also preferred by similar users. Following that recommendation concept, Chen (2005) incorporate context information by computing the similarities between user contexts that replace the similarities between users. As a result, apps are recommended to users when they are preferred by users under similar contexts. Other studies also incorporate context information into conventional recommendation methods. Woerndl, Schueller and Wojtech (2007) implemented a context-aware app recommendation system called play.tools. The author built a cascade hybrid model that first filter apps by content-based information, knowledge-based information and location and then applied collaborative method to further filter the recommendation list. Davidsson and Moritz (2011) recommend apps to new users based on their location and combine location data with collaborative filtering for old users. It is stated in Section 2.2 that conventional recommendation techniques cannot be applied to mobile app recommendation. Although studies mentioned above consider the importance of context information, recommendation techniques are not fit for apps.

For addressing the complex environment of mobile apps, a number of studies integrate context information by leveraging advanced techniques like tensor factorization. Karatzoglou et al. (2012) applied a mathematical model called tensor factorization to solve six dimension matrix where four dimensions are for four types of context information including location, moving, time of day, day of week and two dimensions are for user and item. Shi et al. (2012) use the same technique but improve the previous study by developing a fast learning algorithm for optimizing the recommendation list. The author expanded the data size and showed the scalability of this method. However, the data size is small compared with the large number of apps, users and types of context information in the real app usage environment and scalability is still an issue. In current studies, context information is computed as additional dimension of user contents and items, which limit the number of users, apps, and context information that can be incorporated. Overall, context information besides location and time are rarely considered.

A survey that summarized studies related to app usage prediction and app recommendation indicates current semantic approaches of mining smartphone data are still at initial stages and future research is needed for rigorous and effective semantic analysis (Cao & Lin, 2017). Text mining techniques has been used for extracting user contexts from app reviews and facilitating app recommendation (Kaji et al., 2011). However, Kaji et al. (2011) recommend apps exclusively based on context information and user reviews. In this thesis, I propose a broader conceptual framework for app recommender system that can leverage multiple types of information, including user contexts. Since a large number of people use social network services, textual data can be retrieved from social network platforms such as Twitter, Facebook, LinkedIn, etc. Viewing context information as textual information, there is a possibility of incorporating them into content-based user profile or leveraging them in app categorization schemes to facilitate scalable app recommendation.

2.4 App Classification Schemes

As mentioned before, there are millions of apps released in app stores with substantial new apps being introduced each year. Therefore, selecting a good app classification scheme addressing personal needs and context information becomes a big challenge. An effective categorization scheme can benefit both users and developers by facilitating the understanding and discovery of apps and further facilitating app recommendation (Liu, Song, Baldi, & Tan, 2016). However, existing app classification schemes are limited in achieving any of those benefits. Firstly, categories are mostly labelled by some experts and apps are assigned to one of the categories by app developers (Vakulenko, Müller, & Brocke, 2014). The problem is that categories are subjectively selected without linking to apps. Categorization schemes could be biased due to the subjective views and some apps might not be able to fit in any of the predefined categories. Secondly, categories are designed based on themes but not function (Al-Subaihin et al., 2016). Currently, Google play app store contains 33 categories and apps are categorized based on their most related themes. However, compared to themes, function is a more important factor that affects the fulfillment of user needs. Other limitations like the possibility of being misclassified by app developers, the restriction of being exclusively classified into one category are also mentioned in the literature (Liu et al., 2016).

Due to the limitations summarized above, many extended studies regarding app classification were conducted. Studies are divided into two streams. The first stream is to discover an advanced method to categorize apps automatically based on an existing ontology or classification scheme. Olabenjo (2016) indicated that a substantial number of apps are misclassified because of the carelessness of app developers. The author applied Naïve Bayes to classify app and proved that Naïve Bayes, as one of the supervised machine learning techniques, can be used as a potential method to automate app classification. Zhou (2016) leveraged API files to automatically classify apps. Zhu, Chen, Xiong, Cao and Tian (2014) conducted a textual analysis method, Latent Dirichlet Allocation(LDA), on web knowledge and context information and trained a Maximum Entropy model to classify apps given an app taxonomy of Nokia store. The second stream is to design a novel framework of app classification that outperform the current categorization quality. Vakulenko et al. (2014) applied LDA topic modeling to app description from iTunes app store, uncovering the latent topics incorporated in apps. Al-Subaihin et al.(2016) implemented text analytics to extract features from app description and hierarchically cluster apps into groups with similar features. By sub-clustering apps under each original category of google play and blackberry, fine-grained classification schemes are generated. By changing the number of clusters and evaluating the clustering results, the finest granularities are recorded for each category in app stores. Liu et al. (2016) is another study that aim to create a fine-grained, hierarchical app classification framework. The author also extracted feature from app description and applied Non-Negative Matrix Factorization (NMF) to classify apps into a hierarchical categorization that generated from google ad tree[1]. In summary, most of the studies intend to create a fine-grained, flexible classification with a large number of categories since the current

[1] https://adssettings.google.com/authenticated?hl=en

taxonomies are general and cannot differentiate specific app functions under the same category. However, none of the studies tend to develop or adopt a classification based on personal needs. Moreover, a fine-grained categorization results in a large number of subcategories that make recommendation more complicated. In addition, each app is categorized into an exclusive category whereas the multiple functionalities of apps can correspond to multiple categories.

In order to facilitating app recommendation and link context information to user preference, a course-grained taxonomy based on personal needs is needed. The classification idea of Wang et al.(2014) is similar to my proposed framework in a way that apps are classified to satisfy users' needs. The author classified mobile apps in the "Health & Fitness" category of iTunes App Store for assisting elder people to choose appropriate apps that accord with their needs such as disease prevention or disease management. In my thesis, I expand this idea to adopt a comprehensive, purpose-driven classification of mobile apps. A large number of blogs and white papers provide guidance of app classification. DuckMa (2016) classified apps into 6 main types—lifestyle, social media, utility, game or entertainment, productivity and news. This classification scheme is a simple version of categories in google play, therefore, it is also theme-based. Splendor (2017) categorize apps into 5 needs—apps that solve a problem, apps that have common function, apps that connect you with others, apps that help you get access to the information and apps that benefit you. Though this classification scheme is related to personal needs, it is hard to precisely allocate apps to those categories due to the ambiguous description. In the literature of web search, Rose and Levinson (2004) proposed a framework that classify queries into three categories—informational searching, navigational searching and transactional searching. We can assume that the users search mobile apps based on those three purposes and needs. However, this classification is too broad and simply categorizing apps into these three searching purposes may not facilitate recommendation process extensively.

A study conducted by INSIGHTSNOW, AOL and BBDO (Macki & Draper, 2012), also featured in Harvard Business Review ("How People Really Use Mobile," 2013) presents a comprehensive framework based on the purpose of app usage and the needs of users. In this study, the author conducted interviews, survey and tracked user behavior, summarizing seven types of mobile moments: *Self-expression, Discovery, Preparation, Accomplishing, Shopping, Socializing and Me time*. Explanation and representative apps for each category are summarized in Table 2-1. The study also indicates that apps not only fulfill user needs related to the functions, but also non-intuitive needs. For example, the function of app Amazon is shopping and people are accustomed to associating Amazon with shopping needs. However, people may use Amazon to pass their leisure time by browsing through various merchandise, which may make it correspond to *Me time* as well.

In this thesis, I adopt this framework to address the current gap and limitation of app classification schemes in the context of mobile app recommender systems. Usage purposes of some apps can be entirely different from their themes. An example is Oxford dictionary of English, which is used primarily for learning, but generically classified under the theme of book & reference. Also, functions of apps do not necessarily match the personal needs on all occasions. 'Non-intuitive' needs like passing the time is not related to app functions. For example, Amazon is categorized into *Shopping* in all other function-based frameworks, but can be categorized into *Shopping* and *Me time* in the adopted framework. In contrast to the majority of existing app classification frameworks that are either theme-based or function-based, the adopted framework is able to express all the purposes and satisfy user needs in terms of app usage because it is generated based on what user desire from apps. Moreover, this framework is broader and can be easily linked to user context information. For example, students are more likely to choose apps from *Discovery*, Women tend to do *Shopping* frequently, and businessman need to use apps for *Preparatio*n. As a result, this comprehensive framework of app classification can play a significant role in understanding user needs, integrating context information, and improving the effectiveness of app recommendation extensively.

Table 2-1 : Seven Shades of Mobile (Macki & Draper, 2012)

Categories	Description	Examples
Me time	Relaxing oneself or pass the time	Netflix, Coloring, Face changer
Socializing	Communicating with people	Facebook, Instagram, Viber
Shopping	Seeking or purchasing a product or service	Amazon, eBay, Zara
Accomplishing	Managing life activities	Money manager, Microsoft office
Preparation	Planning for activities	Flights, Trip advisor, Calendar
Discovery	Learning or finding new information	Dictionary, Kindle, Learning English
Self-expression	Expressing personalization and passion	Fantasy football, Wallpaper, Run keeper

Chapter 3 Conceptual Framework and Theoretical Foundation

In this chapter, I outline the conceptual framework for the proposed app recommender system and provide theoretical foundations for the underlying recommendation method.

3.1 Theoretical Foundations of Proposed App Recommendation

Based on my review of the literature, existing mobile app recommender systems are limited mainly in two aspects, namely, the problem of scalability when leveraging context information and the gaps in addressing the needs of users in the app classification schemes. The common method of incorporating context information into app recommendation involves solving a user-app-context multidimensional model where each type of context information is computed as a new dimension (Adomavicius et al., 2005). However, this method is not scalable to include various types of context information due to the increasing complexity of mathematical models with new dimensions and data size. The model can be computationally prohibitive. In order to address this problem, I employ text-mining techniques to leverage context information as well as an app classification scheme based on purposes of app usage and user needs.

Text analysis has been used to solve mobile app retrieval task in which the user explicitly searches apps by keywords (Park, Liu, Zhai, & Wang, 2015). The authors train a dataset containing app description and customer review for computing the degree of relevance between apps and keywords entered by users. It was shown that text analysis facilitates the effectiveness of app retrieval and outperforms conventional methods. Afterwards, the authors conduct a new study working on retrieving apps from users' implicit status text (Park, Fang, Liu, & Zhai, 2016). For instance, when a user adds a status saying 'I am hungry', the intention of eating is semantically analyzed and apps related to food will be displayed on the interface of searching results. Following the lead in these studies, I plan to explore different mechanisms to link context information from social media status texts with the app classification framework. Since the popularity of social network service (SNS) is constantly increasing, substantial social media status texts updated by millions of users can be collected. Those status texts contain a variety of context information that are related to mobile app usage. For example, ‘I am going to New York’ indicates location and ‘I am unhappy’ indicates emotion.

As mentioned in Section 2.4, an effective app classification scheme can benefit both users and developers by facilitating the understanding and discovery of apps and recommendation (Liu et al., 2016). For instance, Wang et al. (2014) assist elders to search apps belonging to “Health & Fitness” by developing a classification scheme. Also, models of predicting user preferences might not identify information of specific apps that are not pre-classified (Zhu et al., 2014). For example, it is relatively practical to target an app category “Social” compared with an app “Facebook”. Therefore, finding ways of integrating user preferences and context information with a novel app classification scheme can improve the effectiveness of mobile app recommendation systems.

Another limitation of current recommender systems is the gap in addressing user needs and app usage behavior. The importance of understanding customer needs are originally summarized in studies related to business management (Patnaik & Becker, 1999). Besides benefiting the entire business design, a comprehensive research of user needs and behaviors is necessary when designing a new technology because the environments of human computer interactions are complicated and dynamic (Oulasvirta, 2005). A behavior model indicates that technology affects people’s behavior when they are sufficiently motivated; in other words, when their needs are satisfied (Fogg, 2009). Due to the importance of user needs and motivation, empirical studies have been conducted to discover needs related to information technology by analyzing quantitative user data (Kankainen & Oulasvirta, 2002). To this end, I adopt an app classification scheme based on needs or purposes of app usage in order to address the existing gap of recommender systems.

As explained in Section 2.4, the classification scheme called “Seven shades of Mobile” is adopted. This framework, which is based on moments of mobile app usage, is developed through an empirical research conducted by three firms. Moments of mobile app usage are divided into seven segments based on the results from survey, interview, and data of user behavior (Macki & Draper, 2012). The seven moments represent seven purposes of app usage and forms of user needs. Those user needs are also theoretically and empirically supported by other studies regarding smartphone user needs or mobile app user needs. Kang and Jung (2014) provides a structure of smartphone user needs, including physiological needs, safety, belongingness, self-esteem and self-actualization, which are originated from Maslow’s hierarchy of needs (Maslow, 1943). Except for safety, other needs can be linked to “Seven shades of Mobile”. In (Kang & Jung, 2014), physiological needs are defined as basic needs giving user convenience and enjoyment, which correspond to *Me time* and *Preparation* categories in seven shades. Belongingness represent the needs of building and sustaining good relationship, which corresponds to *Socialization*. Self-esteem is satisfied when smartphone users have feelings of self-confidence and self-worth, which can be mapped with *Accomplishing* in seven shades. Lastly, self-actualization is linked with *Discovery* because smartphone users have a sense of self-actualization by developing new skills and education. Moreover, those four needs developed in (Kang & Jung, 2014) are empirically validated through surveys in both US and Korean markets. Sun et al. (2017) reviews the literature of mobile app users’ psychological needs and design a hierarchical structure of those needs. In this study, utilitarian needs, low-cost and health are covered by one segment of seven shades called

Accomplishing because those needs are related to life management. Hedonic corresponding to *Me time* can be satisfied by apps related to entertainment. Social needs are defined as interacting with other people, which is also the description of Socializing in seven shades. Cognitive needs and self-actualization indicate apps with functions like searching information, learning knowledge and improving oneself. Therefore, they match with the description of *Discovery*. Overall, user needs summarized in "Seven shades of Mobile" can be mapped with smartphone user needs in other studies by comparing the description of each need.

As for the system implementation, design space and proof-of-concept of context-aware app recommender system has been studied in (Böhmer, Bauer, & Krüger, 2010). The authors state three procedures required for designing the system:1) collecting user-related, app-related and context related information 2) mechanism for relating users with applications 3) user interface. In my proposed framework, the first two procedures are covered. User-related information and contextual information can be implicitly or explicitly collected. App related information are implicitly collected from app stores. Further details are explained in the subsequent subsection. Following the research of (Böhmer et al., 2010), a physical app recommender system called Appazaar are developed and used for testing multiple recommendation techniques (Karatzoglou et al., 2012; Y. Shi et al., 2012). Moreover, the system is applied for analyzing descriptive app statistics, which verifies that app usages are diverse in terms of app categories and context information (Böhmer, Hecht, Schöning, Krüger, & Bauer, 2011).

3.2 Description of Conceptual Framework

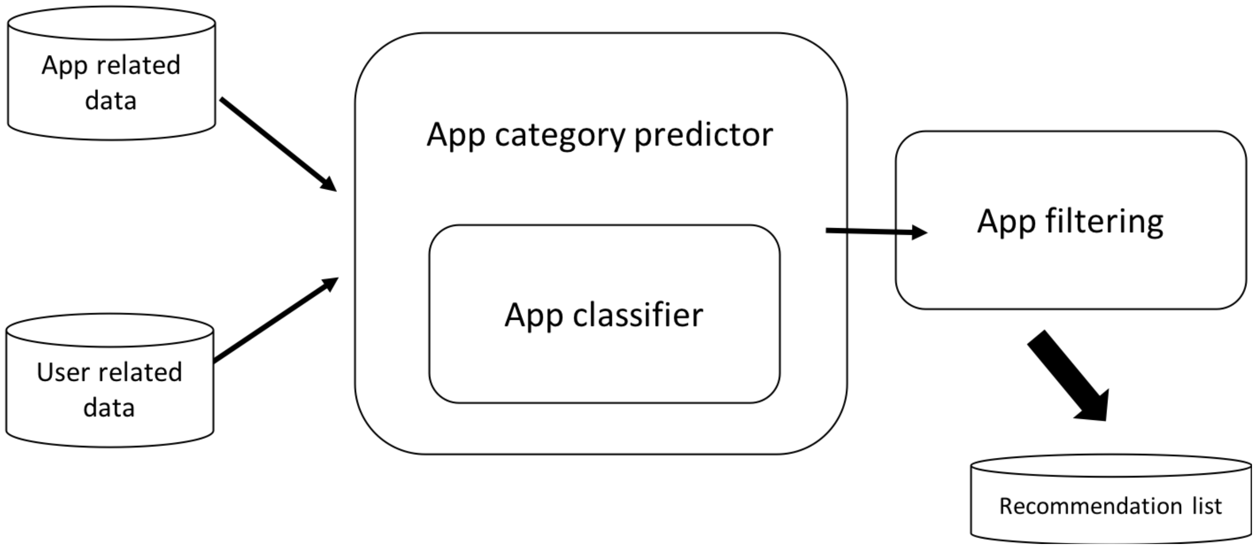


Figure 3-1 : Conceptual Framework

Figure 3-1 displays the proposed conceptual framework of app recommender system. It is mainly composed of two procedures. The first procedure is called app category predictor, which is the primal component of this system. It suggests users' potential preferred app categories based on user-related information. Also, it cascades a procedure called app classifier, predicting app categories based on app-related information. App classifier automatically classifies apps into the classification scheme, "seven shades of mobile" and stores databases of classification results. Given users' preferred app categories generated from App category predictor, App classifier outputs all the apps that belong to those selected app categories. Since several millions of apps in total are available in app stores and the app classification scheme has merely seven categories, there are still substantial apps under desired categories for users to select from. For further filtering the apps, I add the second procedure called App filtering. Ultimately, App filtering output a list of apps that are recommended to users. This study principally focuses on the first procedure, but I propose several options for implementing the second procedure as future research, such as filtering based on randomization, app popularity, or incorporating mechanisms such as those used in Google AdWords in filtering ads from a large number of advertisements that might interest users. Further discussions are outlined in Section 6.2.

Data sources are divided into two portions, namely, user-related information and app-related information. As text analytics is the primary method in this study, data sources have to be collected or converted to a form of textual information. Textual app descriptions are integral part of app-related information. In future research, I plan to add other sorts of information like icons or images. Unlike app-related information, user-related information covers a wide range of data. As in content-based recommendation, user profiles are generated to represent user preferences in this procedure. For instance, when historical data indicates that a user is interested in "Machine Learning", that topic will be added into user profile (Pazzani & Billsus, 1997). User profiles might contain information like demographic, lifestyles, interests and other user-related information. Except for user profile, status texts can be updated by users through social network software (SNS) like Twitter and Facebook. Both user profiles and social media status contain user context information. Since it is difficult to collect all types of user-related information mentioned above, I propose multiple options in terms of user-related information that are displayed in Figure 3-2. This aggregation helps to uncover users' desired app categories from the app classification system.

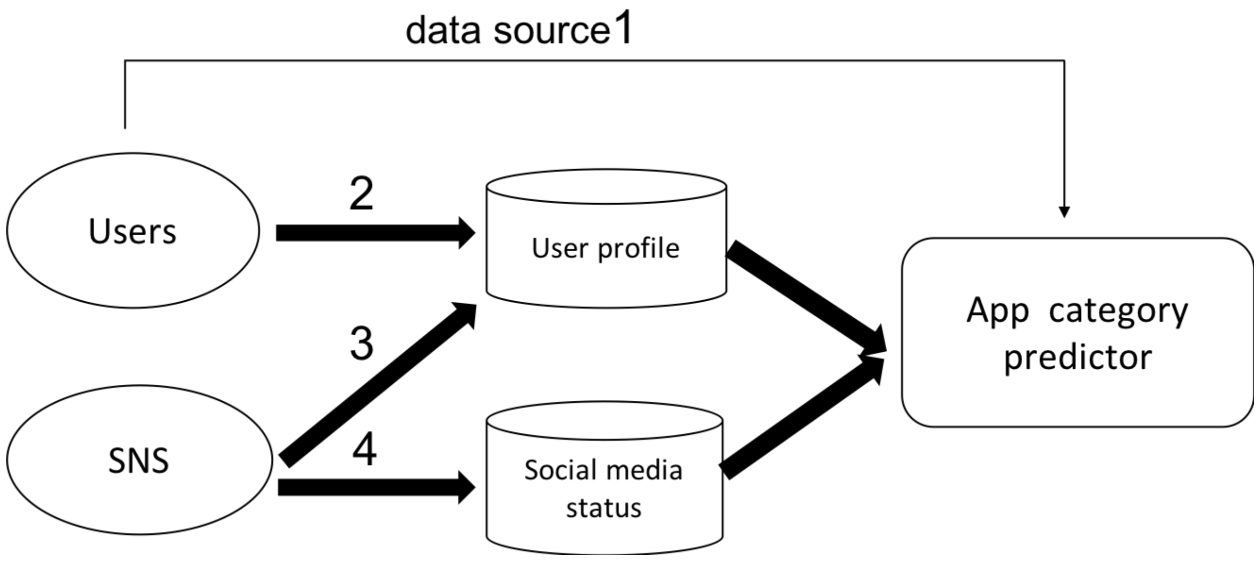


Figure 3-2 : Options of User-related Data Sources

As for the first option, users can explicitly select their desired app categories in the system and those categories can be captured in the adopted app classification scheme. As for the second option, users explicitly upload their profiles to the recommender system. Potentially preferred app categories can be predicted based on user profiles only. For collecting data sources of the third and fourth options, recommender system is connected with SNS. If the users log into the system through social network software (SNS) like Twitter and Facebook that are also installed in devices, then the system will be capable of tracking information updated on SNS by users. Normally, users add their profiles on SNS at the time of signing up a new account. Option three is implicitly collecting user profiles uploaded on SNS by users and option four is implicitly collecting social media status. When multiple data sources are accessible, a new database is developed by merging multiple data sources. For instance, social media status can be added as an extension to the database of user profiles. For each user, all types of user-related information are stored in a form of textual record. Subsequently, users' desired app categories can be predicted based on available information.

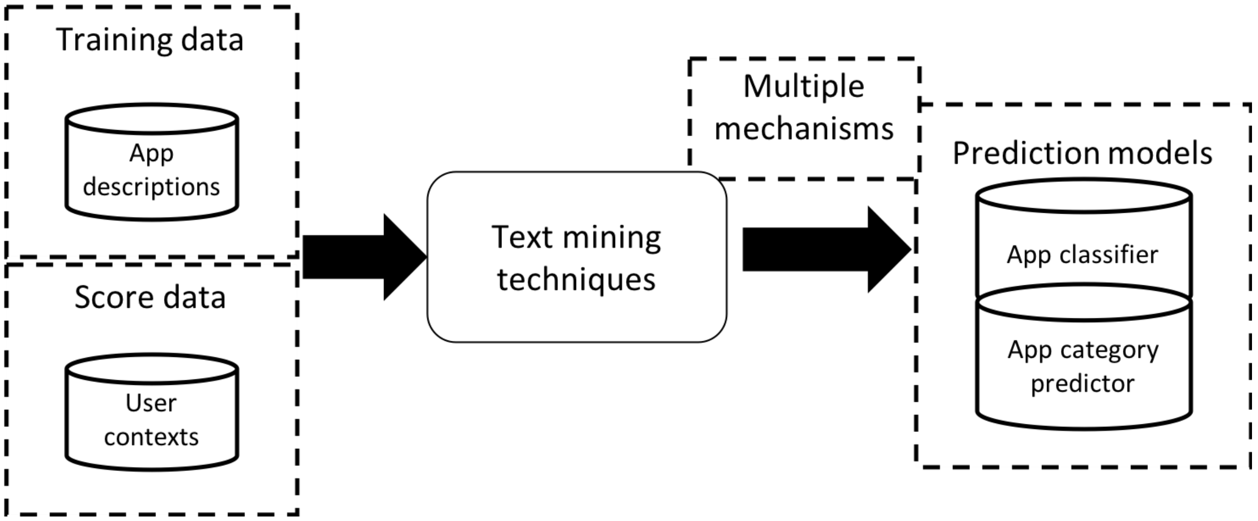


Figure 3-3: Prediction Models

In this conceptual framework, user preferences are expressed by desired app categories in substitution of actual apps. In this study, app descriptions are selected as app-related information and social media status are selected as user-related information or user contexts. For predicting users' desired app categories and assigning mobile apps with app categories, prediction models are needed. In my research, I develop prediction models for implementing the functions of the first procedure. As shown in Figure 3-3, I apply one of the text mining techniques, Latent Semantic Analysis (LSA) to analyze training data including app descriptions. Also, I propose multiple mechanisms to predict user preferences and to classify apps. Specifically, app descriptions are firstly trained to capture the features of app categories and prediction models are developed for predicting app categories based on training data and prediction mechanisms. When applying new textual app descriptions to the model, apps are automatically classified into app categories using App classifier. After that, user contexts like social media status are collected as score data and applied to the prediction models. As a result, user preferences represented by app categories are outputs from the prediction model, which implement the first procedure, App category Predictor. In summary, the same prediction model can implement two functions in parallel: 1) Predicting app categories of apps based on textual descriptions; 2) Predicting user preferences based on social media status.

This conceptual framework provides a scalable approach in terms of leveraging users' context and user behaviors. Besides addressing main problems of traditional mobile app recommender systems, the proposed framework has additional advantages like flexibility. As user-related information can be extracted from multiple data sources, data collection process will not run into any problems when partial information is not accessible. Also, this framework provides a number of mechanisms to predict user preferences represented by app categories. Details of prediction mechanisms are explained in Chapter four. The effectiveness and flexibility of the system can be enhanced by designing many options of data sources, techniques and mechanisms. In the subsequent chapter, empirical experiments are implemented for illustrating the feasibility and effectiveness of the primal process using data from Google Play app store.

Chapter 4 Data and Methodology

This chapter outlines the data collection and the data pre-processing procedures, followed by the textual analysis phases and the corresponding results.

4.1 Data Collection

In order to empirically demonstrate the classification of apps into the adopted classification scheme, 'seven shades of mobile', I collect app data from Google Play app store. Google Play displays a large number of apps online and each app is assigned a link with app-related description. I use a web scraper plug-in of google chrome to crawl the descriptive information of apps displayed on the webpages. In total, 5013 apps from 33 Google Play categories were collected, excluding the game category. The data includes app name, original category, app description, average rating, the number of people who provided rating and other additional information. Original category is the Google Play app store category from which I crawl data. In Google Play, game is further classified into subcategories based on a game classification scheme, and the total number of game apps is particularly large compared to other categories. Thus, game apps represent a complex set of apps requiring an independent classification scheme of its own. As a result, I decided to exclude game apps from the analysis for simplicity. Additional information includes updated date, size, installation numbers, current version, operating system requirements, interactive elements, in-app products, permissions, app developer and app distributor. Table 4-1 presents these details as a sample.

Table 4-1 : Sample Data

Google Category	App Name	App Description
Social	Facebook	Keeping up with friends is faster and easier than ever. Share updates and photos, engage with friends and Pages, and stay connected to communities important to you. Features on the Facebook app include: * Connect with friends and family and meet new people on your social media network * Set status updates & use Facebook emoji to help relay what's going on in your world ....
Shopping	Amazon Shopping	* Customers are able to shop millions of products on any of Amazon's sites around the world from a single app * Quickly search, get product details, and read reviews on millions of products from Amazon and other merchants * Take advantage of 1-Click ordering, customer support, Wish Lists, order tracking, and more ...
Video Players & Editors	YouTube	Get the official YouTube app for Android phones and tablets. See what the world is watching -- from the hottest music videos to what's trending in gaming, entertainment, news, and more. Subscribe to channels you love, share with friends, and watch on any device. ....

4.2 Data Pre-processing

Original categories, average rating, the number of people who provided rating and additional information are collected for implication and supplementary analyses if needed. For implementing the process of app classification by text mining techniques, I keep two variables, namely, app name and app description. I removed records that represent noise for different reasons. In total, four types of noise are identified. To begin with, I detected 275 apps, the description of which are not written in English. In Google Play, an app can belong to more than one category. Therefore, it is possible to collect the same app with the same description multiple times when crawling data by category. After checking all the duplications in our dataset, 609 records are removed. One of the category called Android Wear indicates the apps that are run on the smartwatch but not smart phones. Since the current objective of this research is to facilitate mobile apps recommendation on smartphones, I removed 30 apps that exclusively belong to android wear. Moreover, I removed the apps with no descriptions. Eventually, 4004 records are left. Table 4-2 displays the frequency of each Google play category in my dataset for preliminary experiment.

Table 4-2 : Frequency of Apps Collected from Google Play App Categories

Google categories	Frequency	Google categories	Frequency
Art & Design	135	**Lifestyle**	96
Auto & Vehicles	108	**Maps & Navigation**	190
Beauty	102	**Medical**	203
Books & Reference	161	**Music & Audio**	151
Business	116	**News & Magazines**	77
Comics	101	**Parenting**	118
Communication	169	**Personalization**	198
Dating	105	**Photography**	199
Education	142	**Productivity**	151
Entertainment	60	**Shopping**	68
Events	70	**Social**	120
Finance	86	**Sports**	90
Food & Drink	47	**Tools**	201
Health & Fitness	209	**Travel & Local**	108
House & Home	74	**Video Players & Editors**	210
Libraries & Demo	81	**Weather**	58
Grand Total	**4004**		

4.3 Text Analytics

For assigning a category to each app, and recommending app categories to users, I use one of the common text mining procedures called latent semantic analysis (LSA). Though it is not realistic to analyze texts semantically by understanding the meaning of each sentences, text can be analyzed by computing the co-occurrence patterns of terms across documents because similar documents contain similar occurrence patterns of terms. For example, articles related to information system are more likely to contain words like technology, hardware, software, internet, etc. Latent semantic analysis (LSA), extracts and constructs multiple latent concepts from a corpus containing a large number of documents by computing the co-occurrence patterns of informative terms across documents. Association between term and document is represented by a weighted term- document matrix. Singular-value decomposition (Albright, 2004) is applied to the matrix for reducing the dimension and only the most informative dimensions remain in the model. As a result, latent concepts constructed from the corpus can be interpreted as groups or clusters of the documents. Documents with similar co-occurrence patterns of terms are likely to be grouped into the same clusters (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990). In this research, each app description is considered as a document and app descriptions of all apps combined represent the document corpus. Clusters of documents are interpreted as app categories. Following the guidance of (Chakraborty, Pagolu, & Garla, 2014), I use SAS enterprise miner to implement the text analysis based on LSA. The following three steps outline the process:

Step1: *Text parsing*: The main purpose of text parsing is converting unstructured app descriptions into structured data by tokenizing the entire corpus and implementing natural language processing techniques such as checking grammar, stemming, removing stop words, and creating synonyms. A tokenization process first breaks up all the sentences of the corpus into separate terms separated by space or punctuation. In this research, I tokenize app descriptions and ignore all the symbols, punctuations and pure numbers, which are not useful for app classification. Subsequently, several natural language processing settings are applied to extract informative terms. Specifically, word stems are equated to the root form. For instance, ‘movie’, ’movies’ are seen as the same terms and ‘improves’, ’improving’, ’improved’ are all seen as ‘improve’. Parts of speech of terms are recognized and terms are removed if they belong to unimportant parts of speech that do not affect meanings of the sentences. A number of entities are identified and recorded. displays a list of entities that are identified in the analysis. Among 18 entities, I remove Address, Currency, Internet, Percent, Person and Phone from the analysis because those terms are not relevant to the classification of apps. I keep Date, Time and Location because they represent common context information, which is relevant to my research objective. Furthermore, synonyms of a term are equated to the original term; for instance, “info” are treated as the same with “information”. For eliminating common words and noise, I add a stop list containing extremely frequent terms like ‘the’, ’is’, ’at’, ‘because’, ‘app’, ‘application’ and irrelevant terms like ‘http’. Spelling check was also applied to the app descriptions, but it does not affect the parsing results extensively. Ultimately, all the unique terms that are understandable and useful will be sent to the next procedure.

Step-2: *Text filtering*: Zipf’ s Law states that rank of a term multiplied by the term frequency approximately equals to a constant (Alexander, Johnson, & Weiss, 1998). According to Zipf’s word frequency law in natural language, there are few terms with high frequency and many terms with low frequency, indicating that terms that are neither high nor low frequency are the most informative. The main purpose of text filtering is extracting informative terms and computing a term-document matrix for the next step. After terms are tokenized and filtered in text parsing process, terms are further filtered by controlling the minimum number of documents in which a term should appear. This means, rare terms are removed if they are not appearing frequently in the number of documents that exceed a predefined threshold. Through the method of trial and error, I find that terms should be kept if they appear in at least two documents of the corpus. Finally, 631832 (84%) unique terms are kept for the subsequent analysis and 119521 (16%) are dropped.

Step-3: *Computing term-document matrix*: After the filtering process, all remaining terms and documents are represented in a vector space via the document by term. The raw document by term matrix demonstrates the frequencies that each term appears in each document. The matrix is normally transposed to term by document matrix before proceeding to the analysis. As a large amount of unique terms appear in the corpus even after text parsing and text filtering process and substantial words appear only in a small number of documents, problems of data skewness, high dimensionality and sparseness arise. The problem of skewness is addressed by assigning weights to the frequencies. Final weights are computed as the product of two weights generating from two weighting schemes, namely, frequency weights and term weights, which are also called local weights and global weights. Frequency weights are calculated for each term in each document, while term weights are calculated for each term in the corpus. Log and binary are commonly used options for calculating frequency weights, while entropy, inverse document frequency, mutual information are commonly used options for calculating term weights (Chakraborty et al., 2014,p.107-111; Pincombe, 2004). In this research, I use log to compute frequency weights and entropy to compute term weight because inverse document frequency(idf) is usually recommended for large documents such as academic articles (Jiao, Cornec, & Jakubowicz, 2015). In Section 4.3.2, I use mutual information to calculate term weights because target variables are used in prediction model. Problems of high dimensionality and sparseness are addressed by projecting the weighted term-document frequency matrix into a lower dimensional vector space and will be discussed in the next procedure.

Step-4: *Dimension reduction*: As for the term-document matrix obtained from previous calculation, the number of rows is extremely large compared with columns because there are substantial terms in a corpus compared with the number of documents. A mathematical process called Singular-value decomposition (SVD) is applied to adjust the number of rows, reduce the dimension of term-document frequency matrix, and solve the problem of data sparseness (Albright, 2004). SVD theorem states that any rectangular matrix of real values can be decomposed into the product of three matrices in equation (1).

$$A = U\Sigma V^T \tag{1}$$

When defining the term-document frequency matrix as *A* with *m* terms and *n* documents, the equation is summarized in Figure 4-1.

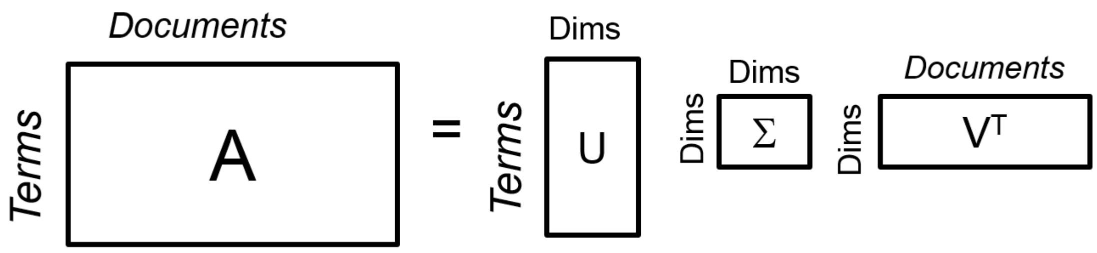


Figure 4-1 : SVD Procedure for Text Analytics (Albright, 2004)

In this equation, *Σ* is a diagonal matrix, the element of which is called ‘singular values’, signifying the importance of SVD dimensions. *T* stands for the transpose of a matrix, and *V* stands for an orthogonal matrix. *U* represents an orthogonal matrix where each term is assigned with a numerical value for each SVD dimension (SVD score). By computing matrix $U^T A$, SVD score is computed for each document based on the SVD score of each term in that document. The total number of dimensions can be controlled by removing the values from the most unimportant dimension to relatively important one. Variances between SVD scores characterize the feature and concept of each document. Documents with similar patterns of SVD scores are likely to be grouped in the same cluster. Ultimately, documents in the corpus are grouped into multiple text clusters, the number of which can be controlled. Besides text clusters, text topics are generated by rotating the $U^T A$ matrix. One document belongs to a single cluster but it is able to belong to multiple topics. As for each cluster or topic, a set of descriptive terms that appear most frequently in those clusters or topics are extracted to describe the features and implicate the results.

In this study, app descriptions are analyzed through the processes of LSA summarized above. For classifying textual app descriptions, an unsupervised exploration is implemented for text clustering as well as topic identification. By employing extensive and systematic exploration of this process, a training data can be generated to facilitate prediction in a supervised setting. This experimentation also includes exploring mechanisms to link context information and user preferences with the app classification scheme. Furthermore, new datasets are applied to prediction models for empirical validations and performance assessments. Finally, social media status information is also applied to prediction models for integrating context information and testing the feasibility of my proposed framework. Figure 4-2 represents the data analysis flow diagram identifying the key phases of my analyses outlined in the subsequent sections.

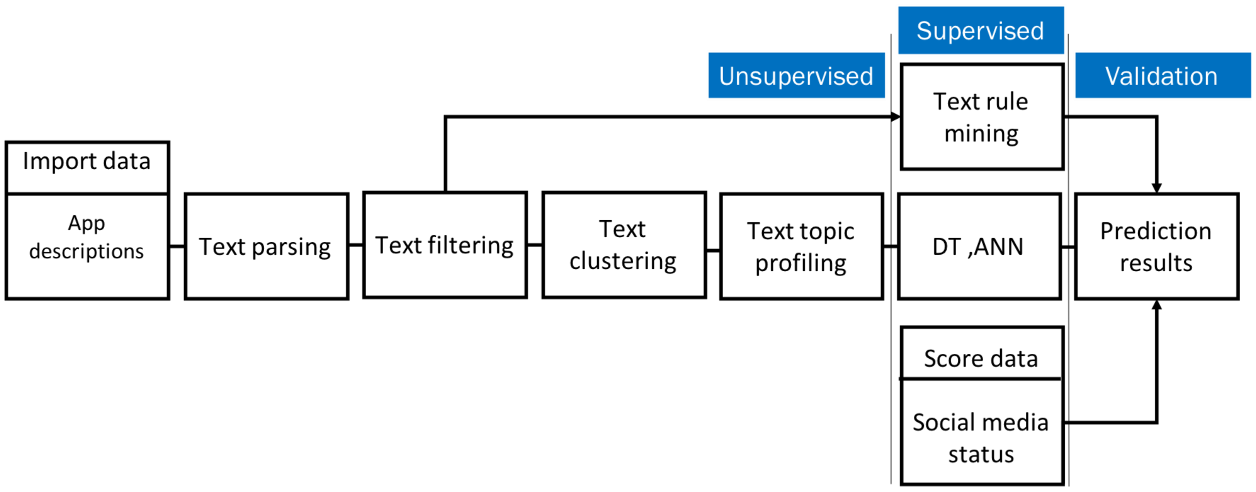


Figure 4-2 : Data Analysis Flow Diagram

4.3.1 Unsupervised Exploration

The objective of the unsupervised learning phase is to develop a deep understanding of the corpus, build a knowledge base for the subsequent prediction steps. Substantial experiments are implemented to detect different app categories based on signals or characteristics of app descriptions. Ultimately, the goal is to map each app with a category from my adopted classification framework. This process addresses the feasibility and the analysis tasks involved in the App Classifier procedure.

During the process of dimension reduction, each app description is assigned with SVD scores for each dimension, and the app descriptions with similar patterns of scores are grouped into the same text clusters or text topics. The experiment starts with exploring pure text clusters and topic identification, followed by repeated examination of remaining cluster results. Through an examination of the app descriptions and descriptive terms that belong to each cluster or topic, text clusters or topics can be manually labelled and linked to the adopted classification scheme. This method generates profiles and establish sufficient knowledge base that can be exploited to prepare training data for supervised app prediction models. Given that this is an unsupervised exploration, it requires an extensive and systematic experimentation. Two primary experimentation factors are the number of clusters and the number of SVD dimensions extracted. As displayed in Table A- 2, descriptive terms and the frequency of each cluster are recorded in each experiment. As seen from the table, semantic meanings of descriptive terms can represent the characteristics of clusters and some of the clusters are extremely relevant to proposed app classification scheme. Furthermore, as displayed in Table A-3, clustering result of apps and the probability of apps belonging to a cluster are recorded. Based on those information, pure clusters are identified when the same apps and the same descriptive terms are consistently grouped together while changing the experimentation factors. A strategy of divide and conquer is used for discovering pure clusters. Once a cluster is considered as a pure cluster, all apps belonging to that cluster are removed from the analysis and the remaining apps are further analyzed until no additional pure clusters are detected.

During the experiment, I also detected clusters that are composed of fewer apps, which can be considered as noise. For instance, some apps can only be installed on android smartwatches and some apps are designed to provide guidance or tips for a specific game. Since those apps are designed for exceptional purposes, I deleted those apps from this analysis. After removing those apps, 4004 observations are left. As a result, eight pure clusters are generated from the entire corpus of app descriptions. After mapping those pure clusters with my proposed classification scheme, five app categories out of seven are extracted. Word clouds created using Tableau visualization are demonstrated below. Each figure represents one type of app category and demonstrates the most frequent terms in that category. Since one category may cover multiple text clusters, frequent terms coming from different clusters are separated by colors.

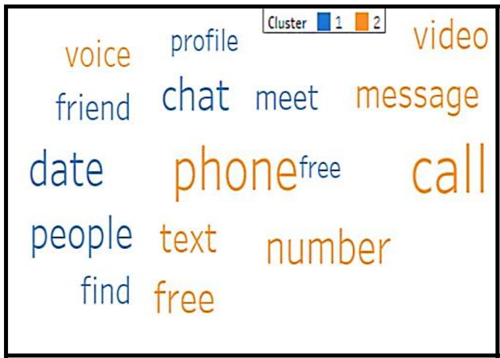


Figure 4-3(1) : Socializing

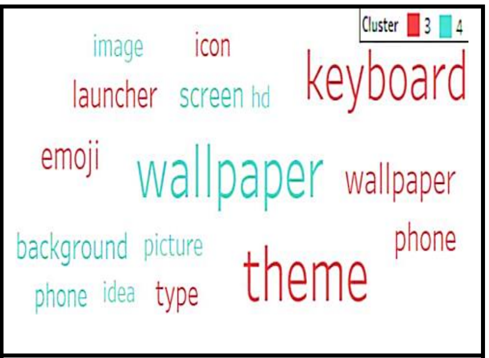


Figure 4-3 (2): Express Oneself

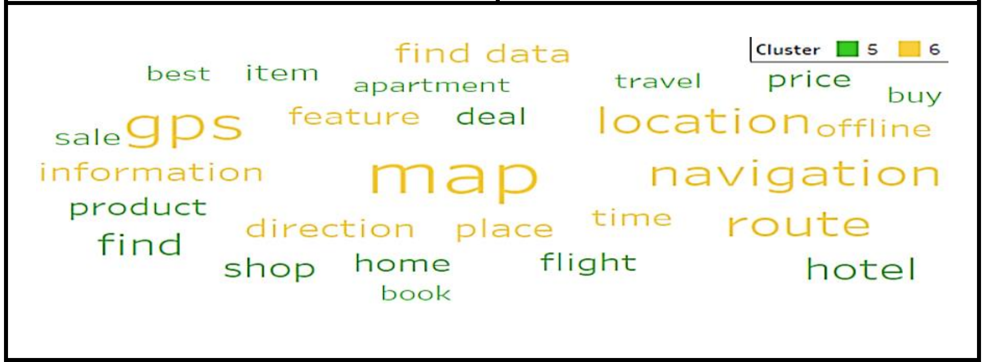


Figure 4-3 (3): Preparation

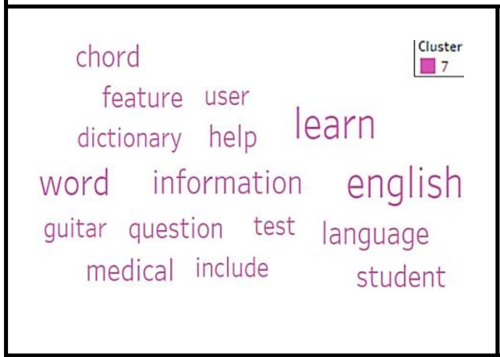


Figure 4-3 (4): Discovery

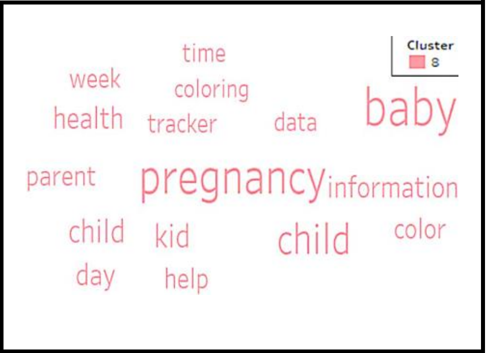


Figure 4-3 (5) : Kids

The first two clusters are both labelled as Socializing because users download those apps for interacting with other people. However, those apps are classified into different text clusters due to the slight distinction on functionalities. Apps from cluster one are generally used for interacting with people online. They can connect with new friends, partners or strangers they have never met before. Facebook and dating apps are examples of this text clusters. Unlike cluster one, apps from cluster two are used for contacting people by phone call or text. In this case, purpose of usage focuses on interacting with acquaintances like friends, business partners, and colleagues because phone number is a piece of private information. Cluster three represents apps with a collection of wallpapers, pictures, decoration ideas, and cluster four represents apps with functions of changing themes, wallpapers, keyboard appearances and emoji. People use apps from both clusters to highlight their personalities and preferences, which correspond to the category of Self-expression.

Apps of cluster five can help people search a variety of information, such as hotel, flight, restaurant, shopping items, among others. Those information is useful for making travel plan or preparing other upcoming activities, therefore, cluster five is labelled as *Preparation*. Though *Shopping* is an independent category of seven shades, shopping apps are completely allocated into text cluster five. Since the data size of shopping is especially small, I remove app category of shopping from seven shades and grouped those shopping apps into *Preparation* for simplicity, because shopping can also be considered as an activity and those apps help user prepare information for shopping. Cluster six are also labelled as *Preparation* because it contains apps that are designed for navigation and making transportation plan.

Cluster seven is summarized as apps for learning skills. People can use apps to learn a new language, prepare a certificate test, look up dictionary or practice a music instrument, functions of which are mapped to *Discovery*. As for the last cluster, it represents apps that are developed for kids to have fun or for parents who desire to parent and take care of their kids. As it cannot be appropriately linked to any categories of seven shades, I create a new app category called Kids. Kids-related apps will be assigned to this category in the following analysis.

Besides apps belonging to those pure clusters, the classification results of other apps are not persistent with changes in the experimental settings. To begin with, music player, video player and video editor are grouped with entertainment apps in some cases and other digital tools in other cases. Reading books are mostly for relaxation but possibly also for discovering new knowledge and information. Weather apps are assigned as assistant tools to support our daily lives but they are also used to plan upcoming activities. More importantly, assistant apps account for the largest portion but the classification of those apps are ambiguous. Some tools like calculator, flash light, online banking are found in independent clusters or grouped with different clusters in different experiments. Those problems are generated due to the contradiction between multi-functionality of apps and exclusiveness of clustering analysis. For addressing these problems, topic analysis is implemented because one app can be classified into multiple topic categories. The results of the topic analysis on the full dataset are summarized in Table 4-3.

Table 4-3 : Results of Topic Analysis

Topic ID	Term1	Term2	Term3	Term4	Term5	Seven Shades
1	date	chat	people	meet	girl	Socializing
2	call	message	number	text	phone	Socializing
3	keyboard	theme	emoji	emoticon	type	Self-expression
4	wallpaper	HD	image	picture	application	Self-expression
5	hotel	shop	deal	search	price	Preparation
6	map	route	navigation	GPS	location	Preparation
7	baby	pregnancy	child	parent	health	Kids
8	word	learn	language	English	dictionary	Discovery
9	news	NHL	TV	team	league	Discovery
10	photo	collage	sticker	editor	maker	Self-expression
11	music	video	player	song	audio	Me time
12	workout	fitness	calorie	exercise	weight	Accomplishing
13	file	battery	junk	protect	security	Accomplishing
14	widget	screen	alarm	clock	flashlight	Accomplishing
15	weather	forecast	widget	wind	temperature	Accomplishing

As seen from the table, the five most frequent terms of each topic are displayed and all the topics are reasonably linked with categories from "seven shades". The first eight topics are consistence with eight pure clusters, which strengthen the reliability of cluster analysis. Apps that do not belong to pure clusters are summarized from topic nine to topic 15. Topic nine was linked with *Discovery* because people not only find new information through learning process but also by checking news, watching TV and seeking sport events. Topic ten was linked to *Self-expression* because mobile users download apps of topic ten in order to make themselves appear better. It contains apps for editing photos like changing photo frames or adding stickers and apps related to makeup and hairstyles. Topic 11 represent apps for entertainment and relaxation like listening music, watching movies, sleeping, which are consistent with the description of *Me Time*. Assistant apps that help people solve specific tasks are all covered from topic 12 to topic 15. Although the classification of assistant apps is not persistent during the experimentation of text clustering, those apps are generally grouped into four categories according to the results of topic identification. Topic 12 represents health-related tools for recording exercises and controlling eating habits. Apps of topic 13 are office tools like email, browser, pdf reader, while apps of topic 14 are digital tools such as widget, flashlight, launcher and system cleaner. The last topic covers weather apps that forecast the weather and temperature in detail all over the world. All assistant tools are linked with *Accomplish* because those apps are used to accomplish specific tasks. The maximum topic weight and minimum topic weight for each topic are summarized in Table A- 4. As seen from the table in Appendix, the maximum topic weights of topic 13 and topic three reach 0.8 and the maximum topic weights for many other topics reach 0.6, which validate the reliability of topic analysis.

In summary, apps of different text topics represent different types of user needs and apps are automatically classified into purpose-driven categories. In addition, a portion of text topics are strongly related to the user contexts. For instance, female is more focused on appearance and often use apps of topic ten. People use apps of topic eight when they desire to learn new skills like programming, spoken languages, among others. Users have a high possibility to download apps of topic 11 while getting bored on the way or killing time at home. Therefore, the classification scheme generated by text mining techniques are purpose-driven and context-related, which match the goal of this research. The frequent terms of each cluster or topic can be treated as signals to build the knowledge base for prediction models.

Topic identification is a process of unsupervised exploration, therefore, I integrate association rule with text mining techniques for validating the reliability of topic analysis. Association rule is originally used to find relationships between transactional variables (Agrawal, Imieliński, & Swami, 1993). However, it has also been used as a method to analyze large textual datasets, for instance, research abstract (Rahman, Sohel, Naushad, & Kamruzzaman, 2010) and research papers (Rekik, Kallel, Casillas, & Alimi, 2018). As explained in (Rahman et al., 2010), the concept of association rule can be applied to text data when frequent terms of documents are considered as transactions. By implementing this method, top five terms extracted from topic analysis are able to generate rules. Taking the first topic as an example, multiple rules are summarized as follow.

Topic 1: date, chat, people, meet, girl
Rule **a. {date} ⇒ {chat}**
Rule **b. {date, chat} ⇒{people}**
Rule **c. {date, chat, people} ⇒{meet}**
Rule **d. {date, chat, people, meet} ⇒{girl}**

Table 4-4 : Examples of Association Rules

Since lift is a function for calculating the interestingness of an association rule (McNicholas, Murphy, & O'Regan, 2008), I use the concept of lift to validate the strong connection between top five terms of the same topic indicating the effectiveness and validity of the topic analysis. Suppose the rule is term A ⇒ term B and P (A) represent the probability of app descriptions containing term A, then P(A, B) represents the probability that two terms appear simultaneously in the same app description, which is called joint probability. In addition, P (A) P (B) represents the probability that two terms appear at the same time under a condition of random events, thus representing independent probabilities. Lift is computed as in equation (2).

$$Lift = \frac{P(A,B)}{P(A)P(B)} \tag{2}$$

The occurrence of term A and term B are independent when P (A, B) = P (A) P (B), in other words, the rule has a lift of one. A lift value greater than one proves that term A and term B have a tendency to appear in the same app description simultaneously because they related to each other but not because of coincidence. Large lift values indicate a strong relationship between terms contained in the rule. When the rules contain four or five terms like rule **c** or **d** in Table 4-4, the independent probability will be an extremely small value, resulting in a large lift value. For addressing that issue, I record the denominator and numerator of lift respectively without taking the ratio and expect the numerator to be predominantly greater than the denominator. In order to visualize the probability clearly, I transform the equation of lift as below.

$$Lift = \frac{|\log(P(A,B))|}{|\log(P(A)P(B))|} \tag{3}$$

After the transformation process, denominators are expected to be greater than numerators. According to the result of topic analysis, the most five frequent terms are recorded for each topic. As five terms can generate 26 rules, 364 rules in total are generated for all 14 topics. In addition, I create five distinct sampling data for validating the consistency of results. Each sample contains 1000 app descriptions, which is one fourth of the entire dataset. I use proportional –stratified sampling (Hirzel & Guisan, 2002), where the original Google categories are the stratification bases. By this method, each sample data covers all the Google categories and the proportion of each Google category is consistent with the population, which avoid the problem of sample bias and represent the characteristics of the entire data. Denominators and numerators of transformed lift are calculated for all the rules generated and repetitive calculations are performed on five sampling data. Figure 4-4 displays plots of partial results.

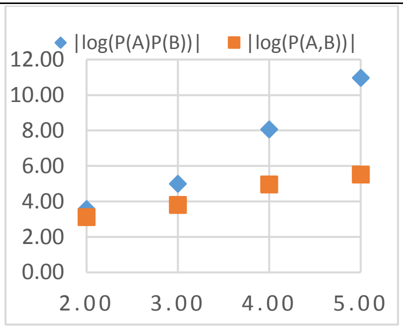
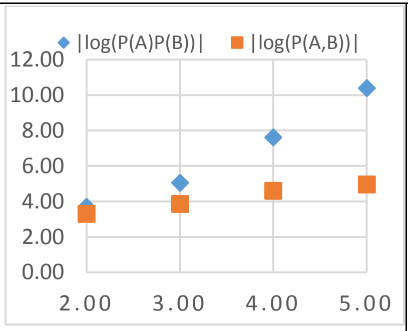
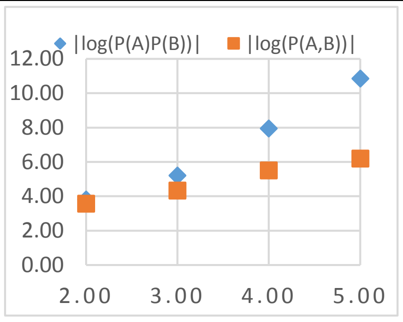
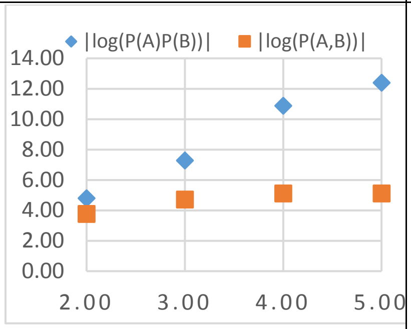
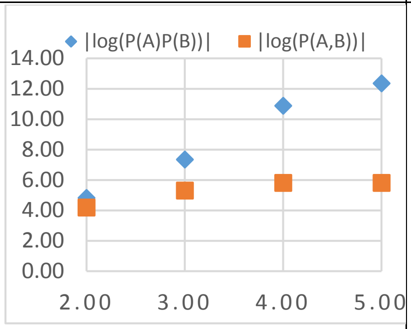
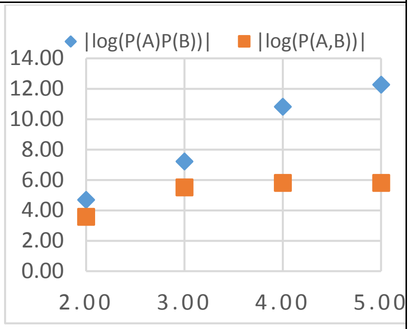
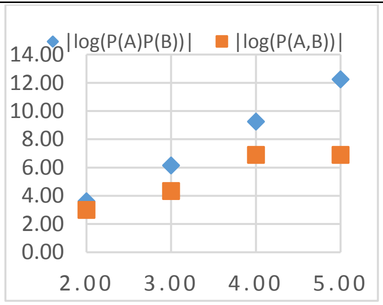
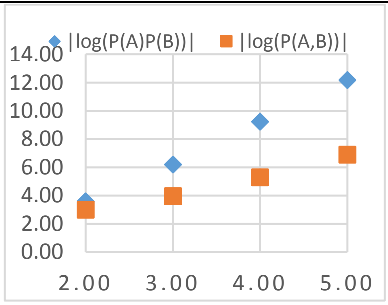
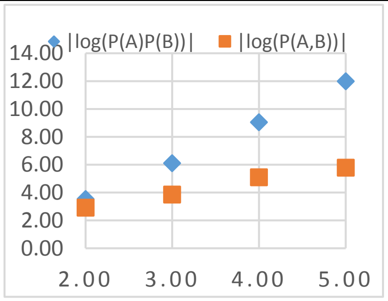
		
(1)Topic 5 sample1	(2)Topic 5 sample2	(3)Topic 5 sample3
		
(4)Topic 9 sample1	(5)Topic 9 sample2	(6)Topic 9 sample3
		
(7)Topic 11 sample1	(8)Topic 11 sample2	(9)Topic 11 sample3

Figure 4-4 : Plots of Denominators and Numerators of Transformed Lift

Among 26 rules generated by top five terms, I selected four rules that have the same patterns with the rules displayed in Table 4-4. X axis represents the number of terms in the rule;

for instance, rule **a.** has two terms, while rule **d.** has five terms. Y axis represents the denominator and numerator differentiated by diamond and square. The first three plots demonstrate the result of topic five, which match with the fifth pure cluster; and the last six plots show the results of topic nine and 11, respectively. As seen from Figure 4-4, denominators are greater than numerators to a great extent in spite of the topic ID and sample number. Besides the partial plots shown above, the remaining data also demonstrate the same result. In conclusion, the top five terms of each topic are strongly related to each other including both the topics that are consistent with pure clusters and the topics that are exclusively extracted in topic analysis. As descriptive terms of topics occur jointly due to their connections but not randomization, the reliability of text topics are testified. After mapping those text topics with app categories of "Seven shades", apps belonging to a topic have a high probability to be classified into the corresponding app category. That process will be formalized and the accuracy will be further tested in the subsequent prediction setting.

4.3.2 **Supervised Learning (Prediction Models)**

Unsupervised exploration testifies the feasibility of automating the app classification task. The performance of this task will be assessed through supervised learning process. According to the summary results of the unsupervised learning phase, apps can be classified into seven categories based on purpose of usage or user needs. The seven categories are *Accomplishing, Discovery, Me Time, Self-expression, Kids, Preparation* and *Socializing*. In this phase, I develop prediction models to assign app categories to apps based on app descriptions. For accomplishing the prediction task, adequate training datasets and prediction mechanisms are required. Training data are generated based on the text analysis steps outlined in the previous unsupervised phase. Frequent terms of each app category are seen as indicators of prediction models. As for prediction mechanisms, I explore multiple methods such as association rule mining, decision trees and neural networks.

Target variables are binary variables indicating whether an app belongs to a given category. If an app is labelled as *Socializing* based on the app description, then the target variable of Socialization will be labelled as one, otherwise, it becomes zero. After text clustering process, each app is classified into a unique text cluster that can be mapped with one of seven app categories. In other words, apps are exclusively classified into "seven shades" classification scheme based on the clustering results. However, those results come from text mining techniques with no guarantee of the accuracy. Furthermore, it is necessary to confirm if one app can be labelled as multiple app categories. As a consequence, I read app descriptions manually and label target variables on the basis of clustering results. The extensive and interactive steps of the unsupervised learning phase was very useful in generating labelled data for the prediction setting. Considering the workload of this process, I took a sample of app descriptions. As mentioned in 4.3.1, I use proportional stratified sampling method where proportion equals to 25 percent. Ultimately, I labelled the app categories of 995 apps out of 4004 and examine the labels at a second time in order to maintain consistency. Positive groups represent apps with a target value of one (Primary target class level), while negative groups represent apps with a target of zero (Secondary target class level). The number of positive apps for each category are summarized as below.

Table 4-5 : Count of Positive Groups (Target=1)

Accomplishing	388
Discovery	234
Me time	218
Self-expression	169
Kids	50
Preparation	167
Socializing	95
Total	995

As seen from Table 4-5, among seven target variables, *Accomplishing* has the largest positive group in which contains approximately 40 percent of the total number of apps. However, other target variables have no more than 25 percent of the apps belonging to positive groups. Consequently, target distributions are imbalanced, which cause a problem of class imbalance that may have an effect on the effectiveness of learning training data (Japkowicz, 2000). As for the prediction mechanism of rule mining, I solve the problem by data preprocessing methods like oversampling the minority class or under-sampling the majority class (Lin, Tsai, Hu, & Jhang, 2017). To ensure all the target variables have relatively equal proportion of positive groups and negative groups, I randomly sample apps belonging to negative groups. After that, training data consists of half of positive groups and half of negative groups for all the target variables.

As explained in Section 4.3.1, rule mining has been used to analyze textual data. Also, rules can be learned to automate the text classification process (Apté, Damerau, & Weiss, 1994). A set of boolean rules indicating whether a term or multiple terms appear in an app description or not can be created to differentiate an app category from the others. As explained in Section 4.3, text parsing and text filtering steps will be conducted on the sample dataset containing 995 app descriptions with labels of app categories. Training data consists of those app descriptions that are tokenized and filtered from the text analytics process and target variables are labelled app categories. For avoiding the problem of overfitting, I split apps into two groups, 50 percent of training data and 50 percent of validation data. In this study, positive apps (i.e., primary target class level) are consistently equally distributed when separating the data. The evaluation criteria are precision and recall. In this prediction model, True Positive (TP) and True Negative (TN) imply correct classifications of an app in Primary and Secondary class level, respectively. On the other hand, False Positive (FP) and False Negative (FN) imply incorrect classifications of an app in Primary and Secondary class level, respectively. The standard metrics for precision and recall are defined in equation (4) and equation (5), respectively.

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{4}$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{5}$$

Both precision and recall are criteria to measure the performance of a prediction model. Precision is the percentage of predicted apps for a given app category that are also labelled as that category while recall is the percentage of apps belonging to a given category that are correctly classified (Apté et al., 1994). Despite the trade-off between precision and recall, a good prediction model performs well on both criteria. F1 score is used to integrate those two methods of evaluation and the formula is displayed in equation (6).

$$F1\ \ score = \frac{2 * \mathrm{Precision} * \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \tag{6}$$

Table 4-6 : Example of Text Rule Mining

	Target Value	Rules	Precision (Train)	Recall (Train)	F1 score (Train)	Precision (Validation)	Recall (Validation)	F1score (Validation)
1	1	learning	1.00	0.12	0.22	0.82	0.08	0.14
2	1	news	0.91	0.26	0.40	0.85	0.25	0.38
3	1	English	0.91	0.36	0.52	0.85	0.29	0.43
4	1	word	0.91	0.46	0.61	0.81	0.33	0.47
5	1	daily	0.92	0.52	0.66	0.79	0.35	0.48
6	1	practical	0.93	0.56	0.70	0.77	0.35	0.48
7	1	ticket	0.94	0.64	0.76	0.79	0.39	0.52
8	1	author	0.94	0.67	0.78	0.77	0.41	0.53
9	0	phone	0.88	0.40	0.55	0.56	0.25	0.34
10	0	save	0.89	0.54	0.67	0.64	0.36	0.46
11	0	screen	0.90	0.63	0.74	0.62	0.43	0.51
12	0	map	0.91	0.67	0.77	0.62	0.43	0.51
13	0	video	0.91	0.72	0.80	0.63	0.44	0.52

Table 4-6 demonstrates an example of rule mining mechanism where target is *Discovery*. Terms appearing in rules of the target are strongly relevant to learning and discovering new information. In contrast, terms appearing in rules where target is zero are not relevant to *Discovery*. In this model, the first boolean rule will be used in priority because it is the best rule for predicting the target. As a consequence, app descriptions containing the term "learning" are classified as *Discovery* and precision and recall after using the first rule are recorded in the first row. Subsequently, the second best rule is selected after removing all descriptions containing the term "learning". Compared with the result of applying one rule, F1 score of applying two rules is greater because the second rule help searching more app of *Discovery*. After rule eight, no more rules can help in finding apps of *Discovery*, therefore, rules are generated for identifying apps that are not belonging to *Discovery*, which also improve the overall performance of the model. The same mechanism will proceed until no more new rules can make significant improvement on the prediction results. Based on the 13 rules, both precision and recall reach relatively high percentages. The final prediction results for seven targets are summarized in Table 4-7 and bar charts are displayed in Figure B- 1. Categories like *Accomplishing, Discovery, Self-expression* demonstrate a relatively high performance on validation data. In contrast, the models of *Kids* and *Socializing* fail to show similar performance.

Table 4-7 : Results of Text Rule Builder

	training		**validation**	
App categories	**Precision**	**Recall**	**Precision**	**Recall**
Accomplishing	86%	73%	69%	58%
Discovery	89%	83%	61%	60%
Me time	96%	75%	75%	49%
Self-expression	96%	88%	87%	81%
Kids	94%	71%	91%	38%
Preparation	98%	75%	77%	51%
Socializing	98%	88%	87%	43%

Besides rule mining mechanisms, I also use decision tree and neural network for predicting app categories. Decision tree and neural network are nonparametric supervised machine learning methods that can be used to train input variables and predict target variables. Unlike rule mining, decision tree and neural network require a large dataset for capturing the difference between app categories. As data sizes are substantially reduced when under-sampling the data in order to keep the balance of target level representation, I use profit-driven prediction models to adjust the class imbalance in substitution for sampling data. Although most of the studies use statistical criterion as main measurement of prediction models, many studies select models with maximal profit (Höppner, Stripling, Baesens, Broucke, & Verdonck, 2017; Zakaryazad & Duman, 2016). After assigning penalty to misclassification cases or profit to correct classifications, the final profit of each model is computed and models with higher profit are preferred. In this study, apps tend to be classified into negative groups because the majority of apps are labelled as zero in each category. Since it is more difficult to identify rare observations, the profit assigned for true positive apps are higher than true negative apps. Specifically, the profit of TP and TN is in inverse proportion to the data size of positive and negative groups, which solves the class imbalance problem. Except for the category *Accomplishing*, prediction models for other categories are generated based on profit.

In this prediction mechanism, the training data contains a variety of input variables. After implementing text analysis on the sample dataset, I add five groups of input variables collected from text clustering and topic analysis. For each app description, the inputs are: 1) SVD scores of each dimension that are generated after step four in Section 4.3; 2) probabilities of classifying the app into each cluster; 3) cluster number of the app based on the ultimate clustering results; 4) topic raw scores, which are rotated SVD scores collected from topic analysis; and 5) topic binary values of the app based on the final results of topic identification. Overall, apps belonging to the same cluster or topic are likely to be classified into the same category. As mentioned in the previous paragraph, better models are selected based on profit and statistical measurements like average square error and misclassification rate are reported. Average square error measures the average of the squares of the differences between target value and predictive value. In this case, the predictive values are the probabilities of allocating apps into target categories. Misclassification rate is the aggregated percentage of FP and FN groups. The standard metrics for average square error and misclassification rate are defined in equation (7) and equation (8), respectively.

$$Average\ Aquare\ Error(ASE) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y_\iota})^2 \tag{7}$$

$$Misclassification\ Rate(MR) = \frac{FP + FN}{TP + TF + FP + FN} \tag{8}$$

After allocating half of the training data to validate the accuracy, prediction results of both decision tree and neural network are summarized in Table 4-8 and bar charts are summarized in Figure B-2 and Figure B- 3. The category *Kids* is excluded from the table because it contains merely 50 positive apps, which are not sufficient for training and validating a reliable model. As for other categories, prediction models perform reasonably well. The majority of models demonstrate a correct classification rate of more than 80 percent. Moreover, the performance of validation data is similar with training data, indicating a high performance when applying new dataset on those models. Generally, neural network performs slightly better than decision trees.

Table 4-8 : Prediction Results

	DT				**ANN**			
	training		**validation**		**training**		**validation**	
App categories	**ASE**	**MR**	**ASE**	**MR**	**ASE**	**MR**	**ASE**	**MR**
Accomplishing	0.10	0.13	0.17	0.23	0.03	0.04	0.16	0.19
Discovery	0.09	0.12	0.13	0.18	0.11	0.14	0.14	0.17
Me time	0.11	0.20	0.13	0.21	0.07	0.09	0.12	0.14
Self-expression	0.07	0.09	0.08	0.10	0.06	0.07	0.07	0.07
Preparation	0.07	0.11	0.08	0.12	0.08	0.09	0.08	0.10
Socializing	0.05	0.07	0.05	0.07	0.04	0.05	0.04	0.05

Concluding from the performances of multiple prediction mechanisms and the similar performances between training and validation datasets, prediction models based on supervised learning procedure indicate the feasibility, flexibility, and scope of text analytics in profiling apps based on textual descriptions. The App classification process can be automated using prediction models after expanding the training data size.

4.3.3 Empirical Validation

In this phase, I conduct further empirical validation on prediction models. Although I use half of the data to train the model and half of the data to validate the accuracy of prediction, validation data are still leveraged to optimize the complexity of the models, for instance, pruning the decision trees. For further validating the performance of supervised learning process, I prepare a new test data that is not used to train or fine-tune the model and does not have any effects on the prediction results. Test data can be considered as new app descriptions that has not been classified. A good prediction result of test data indicates a good performance of app classification in the proposed recommendation framework.

The experiment was conducted on the target category *Accomplishing* for the following reasons. In supervised exploration, apps of this category are not persistently assigned into the same cluster. According to the prediction results in Table 4-8, the performance of the category *Accomplishing* is inferior to other categories. Integrating results of previous experiments, it is relatively difficult to predict apps of this category because it covers a variety of apps that support many aspects of our lives, for instance, health, digital supports, office applications, among others. Apps that are used to manage anything, or accomplish any task can easily be classified in this category. Due to the complexity of this category, empirical validation process is necessary. Also, a large group of apps are labelled as *Accomplishing*, which provide sufficient data for training, validating, testing and avoid the class imbalance problem. Eventually, I split 995 apps into three groups, 40 percent of training data, 40 percent of validation data and 20 percent of test data. Multiple prediction mechanism are implemented and corresponding performance measures are all reported in Table 4-9.

Table 4-9 : Validation Results

Model	Criterion	Data	Value	Criterion	Data	Value
DT	ASE	training	0.12	Precision	training	72%
		validation	0.17		validation	81%
		test	0.20		test	63%
	MR	training	0.14	Recall	training	74%
		validation	0.21		validation	82%
		test	0.27		test	67%
ANN	ASE	training	0.06	Precision	training	94%
		validation	0.15		validation	79%
		test	0.21		test	72%
	MR	training	0.06	Recall	training	91%
		validation	0.16		validation	79%
		test	0.24		test	63%
Text Rule	ASE	training	0.05	Precision	training	98%
		validation	0.09		validation	72%
		test	0.08		test	84%
	MR	training	0.12	Recall	training	79%
		validation	0.35		validation	50%
		test	0.28		test	53%

The objective of this study is to recommend apps that match with users' preferences. As a result, the majority of predicted positive apps should be true positive, in other words, precision is an extremely important evaluation criterion in this setting. For estimating the accuracy of prediction, ASE and MR also play important roles. As for text rule mining, it has the greatest Precision and ASE, whereas the MR and Recall are extremely low. The performance of test data is superior to validation data due to the smaller data size. In terms of ANN and DT, ANN is superior to DT due to its higher precision and lower MR. As seen from the table, decision tree incorrectly classified 27 percent of test data while neural network misclassified 24 percent. In both ANN and DT, performances of test data are slightly inferior to validation data, but the difference between them are small. Overall, MR are less than 30 percent and precision are more than 60 percent for all models. Due to the high performances of prediction models on validation data as well as test data, we can expect reasonably high probability to correctly classify newly released app descriptions.

4.3.4 **Integration of Context Information**

From Section 4.3.1 to Section 4.3.3, multiple mechanisms and validation processes are explored in order to automatically classify apps based on a classification scheme addressing user needs or user preferences. The next step is linking users' context to their preferences, namely, categories of seven shades, which correspond to the first procedure of my conceptual framework called App Category Predictor. As a default setting of inferring users' desired app categories, users can be allowed to explicitly select categories that are displayed on the interface of a recommender system. In this case, training data and prediction models are not necessarily employed. Without considering this exceptional circumstances, the system is designed to identify users' preferred app categories based on their profiles and context information, for instance, status texts updated on SNS. In this phase, I explore multiple mechanisms for validating the feasibility of linking social status texts with the app classification scheme.

A collection of app descriptions is the integral part of training data. It contains not only information related to mobile apps but also users' context, such as "travel", "student", "baby", among others. Through the supervised learning, textual features of each app category are trained and validated, including those contextual terms that are used to describe app categories. Also, prediction models are developed to assigning app categories given textual information. Consequently, we can link app categories to social status texts containing context information after applying prediction models to those status texts. In this empirical experiment, social status texts play a role of score data. I collect status text from multiple sources, such as Facebook status website[2], twitter online datasets[3], twitter advanced search[4]. Also, I use simulated app retrieval queries from (Park et al., 2016). After integrating those data, I simulated 160 status texts, containing 24 texts related to *Discovery*, 33 texts related to *Preparation* and 103 unrelated texts.

Status texts will be applied to text rule-based models and decision trees generated in Section 4.3.2 for predicting whether the user prefers apps from *Discovery* and *Preparation*. As seen from the example displayed in Supervised Learning (Prediction Models) in Table 4-6, rule-based predictions are straightforward. A status text has a high probability to be predicted as one (Primary target class level) when it satisfies the rules that identify the target variable. For example, "I always liked the daily news" is predicted as *Discovery* because it contains the term "news". "I expect work on the map to begin shortly" is predicted as *Preparation* because it contains the term "map". However, rule-based models can be misleading in some cases. Suppose that a user updates a status saying "I do not like the daily news", instead of "I always liked the daily news". In this case, the status may be predicted as *Discovery* as before due to the existence of the term "news". In this case, the system will recommend apps that are not accord with the users' preferences. Another example is demonstrated below.

[2] https://www.dailysmscollection.in

[3] https://data.world/datasets/twitter

[4] https://twitter.com/search-advanced?lang=en

A couple years ago, a very nice young girl was being bullied for wearing a feminist t-shirt in a photo. I offered her words of support.

Though it is a narrative status without indicating user preference, the status may be predicted as *Discovery* because of the existence of the term “words”. As the same term can be used for multiple meanings, part of speeches, and in different situations, a specific term cannot represent the complete meaning of a sentence. Moreover, partial important status information might be ignored because rules do not cover all the frequent terms that feature a category. For instance, “Old Quebec City at night, Travel to Canada” is a representative status of *Preparation* as it indicates that this person is on the way of travelling and need apps to prepare various activities. However, rule-based models do not label the status as target because none of the terms satisfy the rules. Despite the fact that “travel” is an indicator of Preparation, it is not selected to develop a rule and the status texts containing “travel” do not tend to be classified as *Preparation*. According to the prediction results of rule-based models, nine texts are predicted as *Discovery* but merely four of them are actually classified as *Discovery*. Among five status texts that are predicted as *Preparation*, three of them indicate user needs *Preparation*. In summary, rule-based models roughly classify social status texts without addressing every context of term use.

I also use decision trees to predict the users’ preferred app categories based on simulated social status texts. As mentioned in 4.3.2, five groups of variables generated from text clustering and topic analysis are trained to develop decision tree models. The same clustering and topic analysis are implemented on status texts and scores of those five groups of variables are computed respectively. Finally, decision trees assign app categories to each status text based on those scores. Unlike rule-based models, decision trees leveraged advanced text mining techniques for looking deep into status texts. Prediction results are generated after analyzing every single term in the status text but not a few representative terms. Partial examples of positive groups are summarized below.

Discovery: I tried to read a book to my daughter today.

Discovery: I say this as someone STILL annoyed about that early 90's TIME magazine article saying we were all terrible because we weren't doing what boomers thought we should.

Discovery: I'm looking for EXO concert ticket in Malaysia

Preparation: Google Maps should have a feature where you can adjust the distance at which the voice navigation kicks on

Preparation: "If people did travel more the world would be a better place." Two Times travel writers embarked on a road trip across Iceland. See for yourself in virtual reality

Preparation: It was a great day on the parade route!! Another campaign milestone down

The first three pieces of texts indicate that the users are interested in either reading or searching information of events, which accord with the preference of *Discovery*. As for the last three status texts, users show their interests on travelling or outdoor activities, which match the description of *Preparation*. As seen from those simulated status texts, decision tree models capture users' preference of *Discovery* and *Preparation* through textual information. Assuming that users of SNS post their status similar to the texts displayed above, proposed recommender system will identify the user needs and recommend apps belonging to *Discovery* or *Preparation*. Finally, decision tree models correctly identify 14 out of 24 status texts belonging to *Discovery* and 11 out of 33 status texts belonging to *Preparation*.

Compared with rule-based models, decision trees demonstrate superior results and large variations in terms of probabilities of allocating status texts to an app category. Normally, a document is predicted as positive when the probability of belonging to positive groups is greater than 50 percent. By adjusting the threshold of probabilities, additional status texts are allocated to positive groups. After reducing the threshold to 23 percent, 24 texts related to *Discovery* are completely identified. For instance, "Learning full English is my goal by 2019", is correctly predicted as *Discovery*. When setting the threshold to 18 percent, 30 out of 33 texts of *Preparation* are correctly classified. For example, "I am booking a flight to Russia tomorrow", "I can't stop shopping", "Old Quebec City at night, Travel to Canada" and a user profile "Lover of travel" are all classified into positive groups and predicted as *Preparation* after reducing the threshold. However, substantial irrelevant status texts are allocated to positive groups due to the low threshold. In total, 63 additional texts are classified as *Discovery*, but 53 of them are irrelevant. As for *Preparation*, 47 texts are classified into positive groups, whereas 28 of them are unrelated to that category. Overall, lower thresholds result in considerably higher precisions and lower recalls. Therefore, a refined model is needed to address the trade-off between Precision and Recall. Another limitation of current models arises from training data. For example, "Yoga is a way to freedom. By its constant practice, we can free ourselves from fear, anguish and loneliness" is supposed to be *Me time* but is predicted as *Discovery*. Since a few apps provide service of Yoga tutorial, the term "Yoga" appears jointly with terms related to *Discovery* in partial app descriptions. As a result, "Yoga" is considered as an indicator of *Discovery* in decision tree models. The problem is attributed to the small size and poor quality of training data. Therefore, the performance of predictive models can be further testified after expanding the volume and variety of training data.

According to the examples and results discussed above, the feasibility of predicting users' preferred app categories based on status texts is reasonably illustrated. Compared with rule-based prediction, decision trees are more applicable. However, there is still a big gap to improve the quality and effectiveness of the models. Substantial irrelevant status texts are predicted as positive groups and crucial texts are ignored because text clustering and topic analysis are not able to capture the semantic meaning of documents. Ideally, users' desired app categories are recommended to users when they update the social status text containing context information that relate to those app categories.

Chapter 5 Discussion and Implications

In this chapter, I discuss the results in Section 4.3 and their theoretical and practical implications. Google app store classifies mobile apps into 33 categories based on their themes. In this study, I adopt a novel app classification scheme, labelled as, "Seven Shades of Mobile" that addresses user preferences and contextual information. Following the data visualization of (Vakulenko et al., 2014), Figure 5-1 demonstrates the overlap between two classification schemes. Apps are exclusively classified based on two distinct classification schemes. Classification results of Google are generated in 4.1,Data Collection while results of seven shades are generated from the 995 labelled apps in 4.3.2. If one app can be classified into multiple categories, only one dominant category is kept. Figure 5-1 plots the apps classified into each category of seven shades(x axis) and each Google category (y axis). Bubbles show the count of apps belonging to both categories on x and y. The sizes of bubbles are bigger where there are more overlaps. As seen from the figure, the category *Accomplishing* covers apps from a variety of google categories. Apps of *Discovery* are strongly related to Sports, Medical, Education and Book. The category *Kids* can be mapped with Parenting and *Preparation* is mapped with travel and map. Moreover, *Me time* corresponds to Video, Music and Comic, while *Self-expression* correspond to Photography and Personalization. Finally, the majority of the apps for *Socializing* come from Social, Communication and Dating.

Despite overlaps between two classification schemes, we can also observe the differences between them. Figure 5-2 is a bar graph showing frequency of each google category where seven shades are separately represented by different colors. As shown in the graph, apps of one google category correspond to multiple categories of seven shades. For instance, Art and Design covers app from *Accomplishing, Discovery, Kids, Me time and Self-expression*. Consequently, one theme can satisfy multiple types of user needs. It verifies the limitation of Google classification scheme that theme-based classification is not purpose-driven and cannot represent the user needs, which is one of the most important factor in system design (Oulasvirta, 2005). Unlike Google classification scheme, "Seven shades of Mobile" matches my research goal of designing app recommender system.

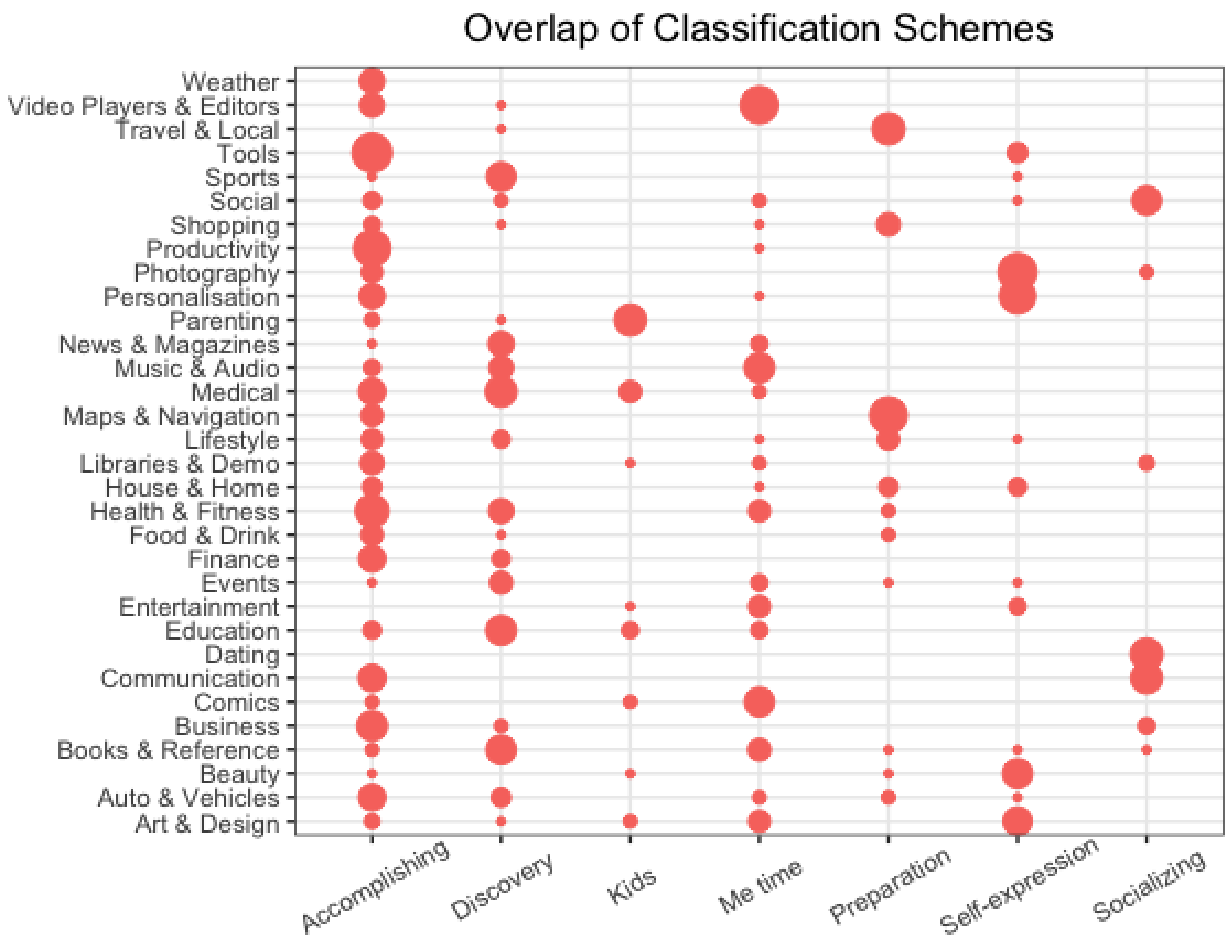


Figure 5-1 : Overlap of Classification Schemes

As mentioned in background and motivation, a large number of apps are released in app stores, indicating that app classification is becoming a challenging task for users. Many studies apply machine learning techniques to automatically classify apps into Google categories (Olabenjo, 2016) and iTunes App Store Categories (Vakulenko et al., 2014). Among those techniques, text analytics is proved to be an effective method to automate app classification process (Al-Subaihin et al., 2016). In this study, I demonstrate the feasibility of textual data analysis in profiling apps and automatically classifying apps into purpose driven categories based on textual descriptions.

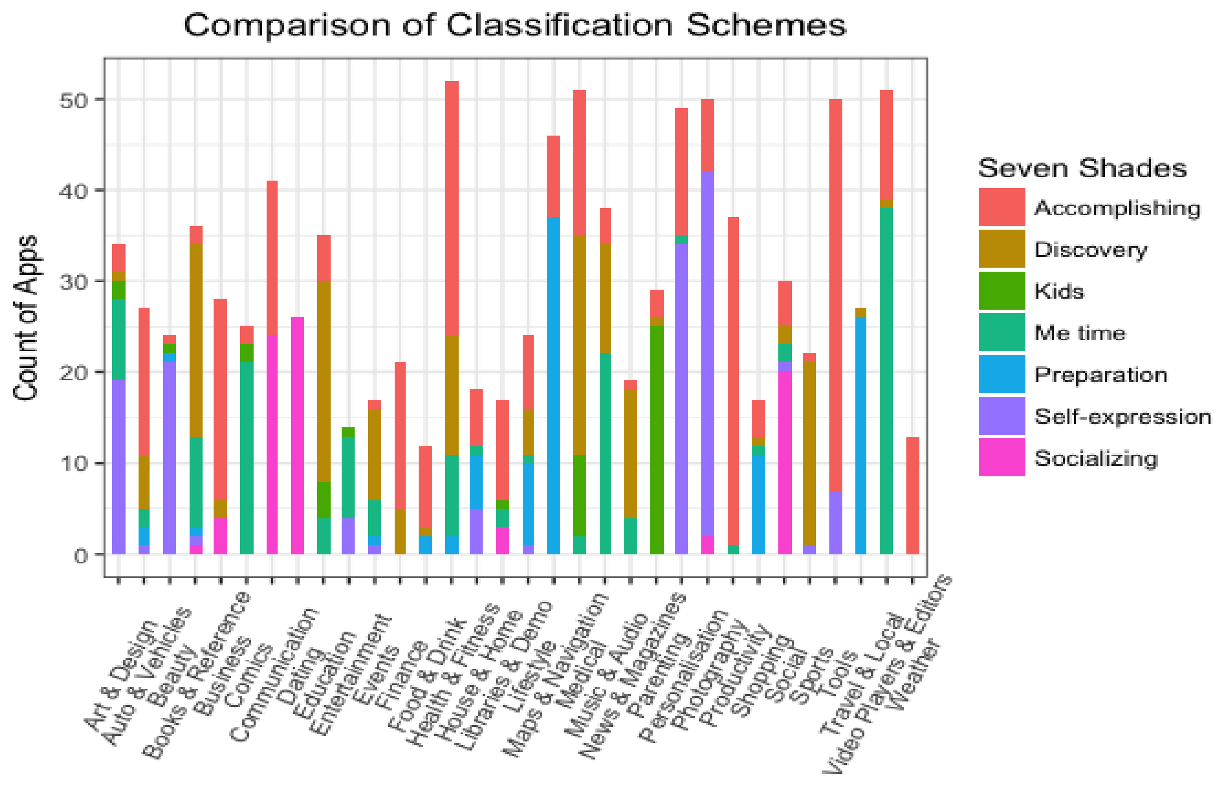


Figure 5-2 : Comparison of Classification Schemes

Through text mining techniques like text clustering and topic identification, app descriptions with similar contents are grouped together. In unsupervised exploration, multiple pure text clusters are extracted and 15 topics are identified. For each topic, terms with high frequencies are defined as descriptive terms, which are used for featuring the topic. By computing Lift, it is verified that terms in the same topic appear jointly with high probabilities because those terms are all strongly related to the text topic. The reliability of topic analysis is also verified because topics are not randomly identified. They are generated based on the co-occurrences of terms. According to the descriptive terms and concrete app descriptions of those topics, it is observed that each topic represents a unique purpose of usage that can be linked to one category of Seven Shades by comparing results of text analytics and definitions of seven shades. In terms of the text analytic results, the categories in seven shades are modified slightly in my experimental setting. Since none of the topics is linked to *Shopping*, it is removed from the seven shades. As a substitute of *Shopping*, an additional category called *Kids* is added for representing apps that are developed for kids or parents. Ultimately, apps are classified into seven categories, including *Accomplishing, Discovery, Me Time, Self-expression, Kids, Preparation* and *Socializing*.

Results of unsupervised exploration shows that text clusters or topics can be reasonably linked to seven shades, but not the original Google categories. Theoretically, seven shades effectively express the user needs or preferences in terms of mobile app usage. Overall, the proposed app classification scheme outperforms Google theme-based classification scheme in addressing user needs and context information. Also, the patterns extracted from unsupervised exploration provides signals and base knowledge for predicting app classification. Taking cluster numbers as an example, apps are assigned with a unique text cluster after extracting textual information from app descriptions. If an app is allocated to a text cluster that is mapped with an app category of seven shades, then the app has high probability to be classified into that category.

Following unsupervised exploration, prediction models are developed under a supervised setting. Textual app descriptions and patterns from unsupervised exploration led to labelled training data. I explore different mechanisms to predict app categories, such as text rule mining, decision trees and neural networks. Although the performances of rule-based models are relatively poor, it is a practical mechanism where implications are straightforward. App categories are predicted based on whether textual descriptions contain terms that satisfy the predefined rules. In contrast to rule based models, performances of decision trees and neural networks are of high quality because text mining techniques are applied to train the data. When applying new app descriptions to prediction models, unclassified apps are allocated to categories of seven shades and the classification process is automated. Since one app might satisfy multiple user needs, an app can be classified into multiple categories.

Besides app classification, prediction models are also used to predict users' desired app categories representing user preferences based on contextual information like social media status. After applying social media status to prediction models, those status texts are allocated to app categories of seven shades. As a result, rule-based models fail to capture user preferences while decision trees precisely predict app categories of partial status texts. Although the overall accuracies of models are not high, performances of models are expected to considerably improve after adding status texts into training data.

After three phases of experimentation, the feasibility and effectiveness of the proposed conceptual framework are testified. Compared with existing recommender systems, I believe that the proposed system is scalable and able to address the complexity of user-item-context information. App classification plays an important role in the system. By classifying apps into purpose-driven categories and summarizing user preferences as those categories, it is scalable to link user preferences with apps despite the huge number of users and apps. The feasibility of automating app classification and linking user preferences with app categories are verified though multiple predicting mechanisms. The recommendation process will be finalized by filtering numerous apps belonging to users' desired app categories. In this study, the method of app recommendation is facilitated by leveraging app classification and contextual information like social media status. Besides contributing to app recommender system, this study demonstrates that text-mining techniques are effective methods for profiling apps and social media status. As for managerial implications, this study provides guidelines for practitioners in customizing products, and services beyond mobile apps. App developers can also reach a wider user base by addressing the needs and preferences of app users.

Chapter 6 Conclusions, Limitations and Future Research

This chapter presents the conclusion of this study. Finally, the chapter ends with the limitations of the study and propose future research directions for addressing those limitations.

6.1 Conclusion

Searching users' desired mobile apps is becoming a challenging task due to the volume, velocity and variety of online information. Therefore, app recommender systems are developed to provide apps automatically based on user preferences. This study proposes a conceptual framework of app recommender system that address the limitations of existing systems. Unlike other systems, proposed system effectively leveraged users' context information and user needs, which are critical in today's app environment and were proved to have strong effects on app usage. The proposed approach is scalable as it leverages app classification schemes based on user needs and context information. In this framework, this study explores different mechanisms to automate app classification process and link app categories with user preferences as well as context information.

According to the results of multiple experiments and validation processes, the proposed app recommendation framework is proved to be feasible and promising. As for theoretical contributions, the proposed conceptual framework addresses limitations of existing context-aware app recommender systems. This study provides a new direction of app recommendation that can become the baseline model for future research. Instead of directly targeting the specific apps, app categories representing user preferences are first selected and after that specific apps under those categories will be pushed to customers. Moreover, The feasibility of textual data analysis in profiling apps and user context is testified. In addition, the study of user needs and app usage can be applied to various research field, such as information system, marketing, socialization, among others. As for managerial contributions, this study potentially benefits both users and developers by improving the effectiveness of app recommendation. This study can guide app developers or marketers to reach a wider user base through a better understanding of user needs and contextual information. Furthermore, apps will be potentially and unconsciously viewed by numerous users through the proposed app recommender system. This study also facilitate app searching and app usage for users because they get access to their desired apps without searching effort. Finally, the recommendation idea is not limited to apps but can be applied to other products or services where context information and user needs play an important role. For example, recommendations of Amazon online products also depend on user contexts and purposes of usage. After adopting a classification scheme that address user needs and context information, the proposed recommendation process can be used by predicting the users' desired product categories based on context information.

6.2 Limitations and Future Research

This study has limitations in three aspects. Firstly, the problem of data size exists throughout multiple phases of this study. Many app classification studies collect more than 10,000 apps (Al-Subaihin et al., 2016; Olabenjo, 2016). Considering the data size of existing studies and the large number of apps that has been released in app stores, a data size of 4000 textual app descriptions is relatively small. Specially, game apps are removed from this study for simplicity and will be included in future research. Web scraper is an extension of Google Chrome browser that is developed for crawling data on webpages. I crawled app descriptions through this tools but it cannot get access to the apps that are not displayed on webpages. For increasing the number of data sample, a computational software linking with API of app stores is favorable. The same data size problem exists in supervised learning and empirical validation. Prediction models of app classification are developed on a basis of 995 sample data because only 995 apps have class variables manually labelled following the topic cluster and topic analysis exploratory phase. For addressing the data imbalance problem, samples taken in rule-based models are smaller than 995. Furthermore, additional social media status can also be collected for further testing the effectiveness of the models. Overall, conclusions drawn from this study is based on a relatively small data samples. A small data size might arise a problem of sampling bias because a portion of data cannot represent the entire population. In future studies, a large number of apps and social media status can be collected and to strengthen the experimental results and conclusions drawn.

Another aspect of limitation is related to methodology. The main research methodology of this study is a commonly used text mining technique called LSA. Besides LSA, Latent Dirichlet Allocation (LDA) has also been applied to text analytics in substantial studies (Vakulenko et al., 2014). Different from LSA, LDA is a probability based model in which probability distributions are computed (Bergamaschi & Po, 2014). LDA can also be included in the analysis and compared with LSA. In addition, other advanced natural language processing techniques can be leveraged to analyze semantic meaning of social media status. A common limitation of text analysis is the lack of quantitative validation in unsupervised exploration. The judgement of pure clusters, linkages between clusters and seven shades categories are open to subjective views. Also, target variables of prediction models are manually labelled. As a future research direction, the effects of subjectivity can be alleviated by collaboration or intervention of other researchers in the process.

As for prediction mechanisms, rule mining, decision trees and neural networks are selected. Besides those methods, other machine learning techniques like Naïve Bayes classification (Olabenjo, 2016) can also be implemented. In rule-based prediction models, majority cases of target variables are sampled in order to keep the balance of positive groups and negative groups. Although the sampling method is random under-sampling, a combination of multiple sampling methods are proved to be more effective (Chawla, Bowyer, Hall, & Kegelmeyer, 2002).

“Seven shades of mobile” is selected as the basic app classification scheme because it addresses both user needs and context information in terms of app usage. However, it is not a unique option and other classification schemes with same functions can be adopted. In unsupervised exploration, 15 text topics in total are identified and each topic represent a specific purpose of app usage. Though 15 text topics are linked to categories of seven shades, we can also define each topic as an app category. In this way, an alternative app classification scheme with 15 categories is generated. Besides the variety of app classification scheme, training data should also be integrated with various attributes. Currently, training data is exclusively comprised of app descriptions. Additional app-related attributes can be incorporated for predicting the app categories, such as icons, customer reviews, among others. More importantly, contextual information or other user-related information should also be included as partial training data. In this study, prediction models are not only developed for app classifications but also for capturing user preferences. For instance, social media status texts are applied to the model for predicting users’ desired app categories. Performances of the models are limited because social media status are distinct type of data sources from training data, namely, app description. This problem can be solved by adding social media status texts as a part of training data. Moreover, other user-related information can be considered, such as user profiles, demographics, among others.

The last aspect of limitation is related to the implementation of conceptual framework. I propose a conceptual framework of app recommender system without developing a real recommender engine. The feasibility and effectiveness of app recommendation are tested on a simulation-based environment. However, the performance of recommender system and the advantages of this recommendation idea need to be proved based on real users. As a future research direction, a physical system can be developed and tested with real users to enhance the practical value of the proposed framework. In addition, as mentioned in Section 3.2, the last procedure of the proposed conceptual framework of the system, App Filtering, is not covered in this study. Given an app category preferred by users, there can still be thousands of apps belonging to this category for users to select from. App Filtering is added as the final procedure for further filtering apps and generating recommendation lists with a number of specific apps. Apps can be filtered based on their popularities or rankings in the app store. However, ranking frauds has been detected in studies (Zhu et al., 2013). An effective app recommender system should raise the chances for users to view a wide range of useful apps, but not only highly ranked apps. As for filtering methods, I propose to leverage the mechanisms used in Google AdWords for rotating specific ads.

Google AdWords displays advertisements on webpages when users explicitly search keywords that are relevant to the ads (Guerini, Strapparava, & Stock, 2010). Suppose that a user search "app recommender system" on Google search engine, ads of app recommendation will be shown on the side of webpages. Considering the large number of ads released by marketers, Google AdWords need to make decisions regarding the ads to be displayed within the same ads group. According to the official blog of Google, AdWords provide two options for ad rotation, namely, rotating ads indefinitely and optimizing best performing apps (Google, 2017). As for optimization, Google design algorithms to prioritize the ads with best performance evaluated by criteria like click through rate and conversion rate (Guerini et al., 2010). In contrast to optimization, Google rotate apps evenly without considering the performances. Adds rotation is important because adds with higher clicks are not often preferred by users depending on specific contexts (Bucea-Manea-Tonis, 2012).

Learning from Google AdWords, I also propose two options for filtering apps within the same app category. The first method is called randomization, which randomly select apps. The second method is optimization, which chooses apps based on scores calculated by algorithms. I propose multiple criteria to select apps that have higher probability to be preferred by users. First of all, apps are filtered based on the actual usage rate. Apps are more likely to be added in the recommendation list when they are frequently downloaded and used through my proposed app recommender system. Secondly, apps can be chosen based on the similarity between app-related information and user-related information. For instance, if textual app descriptions have substantial terms that jointly appear in social media status updated by that user, then those apps tend to be added in the recommendation list. Although the ideas of app filtering are organized, specific techniques and algorithms have not been studied. Moreover, the effectiveness and feasibility of filtering methods are not empirically verified. As a future research direction, empirical validations can be implemented on App filtering.

References:

Adomavicius, G., Sankaranarayanan, R., Sen, S., & Tuzhilin, A. (2005). Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach. *ACM Transactions on Information Systems (TOIS)*, *23*(1), 103–145.

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*(6), 734–749.

Agrawal, R., Imieliński, T., & Swami, A. (1993). Mining Association Rules Between Sets of Items in Large Databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data* (Vol. 22, pp. 207–216). ACM.

Albright, R. (2004). Taming Text with the SVD. In *SAS Institute Inc.* Cary, NC.

Alexander, L., Johnson, R., & Weiss, J. (1998). Exploring Zipf's Law. *Teaching Mathematics and Its Applications: An International Journal of the IMA*, *17*(4), 155–158.

Al-Subaihin, A. A., Sarro, F., Black, S., Capra, L., Harman, M., Jia, Y., & Zhang, Y. (2016). Clustering Mobile Apps Based on Mined Textual Features. In *Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement* (pp. 38:1–38:10). ACM.

Apté, C., Damerau, F., & Weiss, S. M. (1994). Automated Learning of Decision Rules for Text Categorization. *ACM Transactions on Information Systems (TOIS)*, *12*(3), 233–251.

Balabanović, M., & Shoham, Y. (1997). Fab: Content-based, Collaborative Recommendation. *Communications of the ACM*, *40*(3), 66–72.

Barnard, L., Yi, J. S., Jacko, J. A., & Sears, A. (2007). Capturing the Effects of Context on Human Performance in Mobile Computing Systems. *Personal and Ubiquitous Computing*, *11*(2), 81–96.

Basu, C., Hirsh, H., & Cohen, W. (1998). Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence* (pp. 714–720).

Bergamaschi, S., & Po, L. (2014). Comparing LDA and LSA Topic Models for Content-Based Movie Recommendation Systems. In *Web Information Systems and Technologies* (pp. 247–263). Springer, Cham.

Billsus, D., & Pazzani, M. (1998). Learning Collaborative Information Filters. In *Proceedings of the Fifteenth International Conference on Machine Learning* (Vol. 98, pp. 46–54).

Böhmer, M., Bauer, G., & Krüger, A. (2010). Exploring the Design Space of Context-aware Recommender Systems that Suggest Mobile Applications. In *2nd Workshop on Context-Aware Recommender Systems* (Vol. 5).

Böhmer, M., Hecht, B., Schöning, J., Krüger, A., & Bauer, G. (2011). Falling Asleep with Angry Birds, Facebook and Kindle: A Large Scale Study on Mobile Application Usage. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services* (pp. 47–56). ACM.

Breese, J. S., Heckerman, D., & Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (pp. 43–52). Morgan Kaufmann Publishers Inc..

Bucea-Manea-Tonis, R. (2012). Search Engine Optimization in the current socio-economic challenges. *Review of Applied Socio-Economic Research*, *3*(1), 35–42.

Burke, R. (1999). Integrating Knowledge-based and Collaborative-filtering Recommender Systems. In *Proceedings of the Workshop on AI and Electronic Commerce* (pp. 69–72).

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

Cao, H., & Lin, M. (2017). Mining smartphone data for app usage prediction and recommendations: A survey. *Pervasive and Mobile Computing*, *37*, 1–22.

Chakraborty, D. G., Pagolu, M., & Garla, S. (2014). *Text Mining and Analysis: Practical Methods, Examples, and Case Studies Using SAS*. SAS Institute.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Chen, A. (2005). Context-Aware Collaborative Filtering System: Predicting the User's Preference in the Ubiquitous Computing Environment. In *Location- and Context-Awareness* (Vol. 3479, pp. 244–253). Springer, Berlin, Heidelberg.

Chen, G., & Kotz, D. (2000). *A Survey of Context-Aware Mobile Computing Research* (Technical Report No. TR2000-381). Hanover: Dartmouth College, Computer Science. Retrieved from http://www.cs.dartmouth.edu/reports/TR2000-381.ps.Z

Costa-Montenegro, E., Barragáns-Martínez, A. B., & Rey-López, M. (2012). Which App? A recommender system of applications in markets: Implementation of the service for monitoring users' interaction. *Expert Systems with Applications*, *39*(10), 9367–9375.

Davidsson, C., & Moritz, S. (2011). Utilizing Implicit Feedback and Context to Recommend Mobile Applications from First Use. In *Proceedings of the 2011 Workshop on Context-awareness in Retrieval and Recommendation* (pp. 19–22). ACM.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407.

Dey, A. K. (2001). Understanding and Using Context. *Personal and Ubiquitous Computing*, *5*(1), 4–7.

Dey, A. K., Abowd, G. D., & Salber, D. (2001). A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-aware Applications. *Human-Computer Interaction*, *16*(2), 97–166.

DuckMa. (2016, November 9). The 6 Main Types of Mobile Apps. Retrieved October 27, 2017, from http://duckma.com/mobile-app-categories/

eMarketer. (2017). Average Time Spent per Day with Mobile Internet Among US Adults, In-App vs. Mobile Web, 2015-2019 (hrs:mins) - eMarketer. Retrieved November 28, 2017, from http://www.emarketer.com/Chart/Average-Time-Spent-per-Day-with-Mobile-Internet-Among-US-Adults-In-App-vs-Mobile-Web-2015-2019-hrsmins/206443

Fogg, B. (2009). A Behavior Model for Persuasive Design. In *Proceedings of the 4th International Conference on Persuasive Technology* (p. 40). ACM.

Girardello, A., & Michahelles, F. (2010a). AppAware: Which Mobile Applications Are Hot? In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services* (pp. 431–434). ACM.

Girardello, A., & Michahelles, F. (2010b). Explicit and implicit ratings for mobile applications. In *GI Jahrestagung (1)* (pp. 606–612).

Google. (2017, August 29). Introducing a better, simpler ad rotation. Retrieved July 18, 2018, from https://adwords.googleblog.com/2017/08/introducing-better-simpler-ad-rotation.html

Guerini, M., Strapparava, C., & Stock, O. (2010). Evaluation Metrics for Persuasive NLP with Google AdWords. In *Proceedings of the International Conference on Language Resources and Evaluation*. LREC.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75–105.

Hirzel, A., & Guisan, A. (2002). Which is the optimal sampling strategy for habitat suitability modelling. *Ecological Modelling*, *157*(2), 331–341.

Höppner, S., Stripling, E., Baesens, B., Broucke, S. vanden, & Verdonck, T. (2017). Profit Driven Decision Trees for Churn Prediction. *ArXiv:1712.08101*. Retrieved from http://arxiv.org/abs/1712.08101

How People Really Use Mobile. (2013, February). *Harvard Business Review*, 30–31.

Japkowicz, N. (2000). The Class Imbalance Problem: Significance and Strategies. In *Proceedings of the 2000 International Conference on Artificial Intelligence* (pp. 111–117).

Jiao, Y., Cornec, M., & Jakubowicz, J. (2015). An entropy-based term weighting scheme and its application in e-commerce search engines. In *International Symposium on Web Algorithms*. Deauville, France. Retrieved from https://hal.archives-ouvertes.fr/hal-01171138

Kaji, K., Yano, M., & Kawaguchi, N. (2011). App.Locky: Users' Context Collecting Platform for Context-aware Application Recommendation. In *Proceedings of the 2nd International Workshop on Ubiquitous Crowdsouring* (pp. 29–32). ACM.

Kang, S., & Jung, J. (2014). Mobile communication for human needs: A comparison of smartphone use between the US and Korea. *Computers in Human Behavior*, *35*, 376–387.

Kankainen, A., & Oulasvirta, A. (2002). Design Ideas for Everyday Mobile and Ubiquitous Computing Based on Qualitative User Data. In *Universal Access Theoretical Perspectives, Practice, and Experience* (pp. 458–464). Springer, Berlin, Heidelberg.

Karatzoglou, A., Baltrunas, L., Church, K., & Böhmer, M. (2012). Climbing the App Wall: Enabling Mobile App Discovery Through Context-aware Recommendations. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management* (pp. 2527–2530). ACM.

Lang, K. (1995). NewsWeeder: Learning to Filter Netnews. In *Proceedings of the 12th International Conference on Machine Learning* (Vol. 10, pp. 331–339).

Lin, J., Sugiyama, K., Kan, M.Y., & Chua, T.-S. (2013). Addressing Cold-start in App Recommendation: Latent User Models Constructed from Twitter Followers. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 283–292). ACM.

Lin, W.C., Tsai, C.F., Hu, Y.H., & Jhang, J.S. (2017). Clustering-based undersampling in class-imbalanced data. *Information Sciences*, *409–410*, 17–26.

Liu, X., Song, H. H., Baldi, M., & Tan, P. N. (2016). Macro-scale mobile app market analysis using customized hierarchical categorization. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications* (pp. 1–9). IEEE.

Macki, A., & Draper, V. (2012). *Seven Shades of Mobile: The Hidden Motivations of Mobile Users* (Mobile Research). Retrieved from http://www.ana.net/miccontent/show/id/s-mob-feb13-insightsnow

Maslow, A. H. (1943). A Theory of Human Motivation. *Psychological Review*, *50*(4), 370–396.

McNicholas, P. D., Murphy, T. B., & O'Regan, M. (2008). Standardising the lift of an association rule. *Computational Statistics & Data Analysis*, *52*(10), 4712–4721.

Olabenjo, B. (2016). Applying Naive Bayes Classification to Google Play Apps Categorization. *ArXiv:1608.08574*. Retrieved from https://arxiv.org/abs/1608.08574

Oulasvirta, A. (2005). Grounding the Innovation of Future Technologies. *Human Technology: An Interdisciplinary Journal on Humans in ICT Environments*, *1*(1), 58–75.

Pan, W., Aharony, N., & Pentland, A. (2011). Composite Social Network for Predicting Mobile Apps Installation. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence* (pp. 821–827). San Francisco, California: AAAI Press.

Park, D. H., Fang, Y., Liu, M., & Zhai, C. (2016). Mobile App Retrieval for Social Media Users via Inference of Implicit Intent in Social Media Text. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (pp. 959–968). ACM.

Park, D. H., Liu, M., Zhai, C., & Wang, H. (2015). Leveraging User Reviews to Improve Accuracy for Mobile App Retrieval. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 533–542). ACM.

Patnaik, D., & Becker, R. (1999). Needfinding: The Why and How of Uncovering People's Needs. *Design Management Journal (Former Series)*, *10*(2), 37–43.

Pazzani. (1999). A Framework for Collaborative, Content-Based and Demographic Filtering. *Artificial Intelligence Review*, *13*(5–6), 393–408.

Pazzani, M., & Billsus, D. (1997). Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning*, *27*(3), 313–331.

Pincombe, B. (2004). *Comparison of Human and Latent Semantic Analysis (LSA) Judgements of Pairwise Document Similarities for a News Corpus*. DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION SALISBURY (AUSTRALIA) INFO SCIENCES LAB. Retrieved from http://www.dtic.mil/docs/citations/ADA427585

Rahman, C. M., Sohel, F. A., Naushad, P., & Kamruzzaman, S. M. (2010). Text Classification using the Concept of Association Rule of Data Mining. *ArXiv:1009.4582*. Retrieved from http://arxiv.org/abs/1009.4582

Rekik, R., Kallel, I., Casillas, J., & Alimi, A. M. (2018). Assessing web sites quality: A systematic literature review by text and association rules mining. *International Journal of Information Management*, *38*(1), 201–216.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (pp. 175–186). ACM.

Rose, D. E., & Levinson, D. (2004). Understanding User Goals in Web Search. In *Proceedings of the 13th International Conference on World Wide Web* (pp. 13–19). ACM.

Shardanand, U., & Maes, P. (1995). Social Information Filtering: Algorithms for Automating “Word of Mouth.” In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 210–217). ACM Press/Addison-Wesley Publishing Co.

Shi, K., & Ali, K. (2012). GetJar Mobile Application Recommendations with Very Sparse Datasets. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 204–212). ACM.

Shi, Y., Karatzoglou, A., Baltrunas, L., Larson, M., Hanjalic, A., & Oliver, N. (2012). TFMAP: Optimizing MAP for Top-n Context-aware Recommendation. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 155–164). ACM.

Soboroff, I., & Nicholas, C. (1999). Combining Content and Collaboration in Text Filtering. In *Proceedings of the IJCAI Workshop on Machine Learning for Information Filtering* (Vol. 99, pp. 86–91).

Splendor. (2017, April 6). Categorizing Popular Mobile Applications:5 Ideas to Consider in the Ideation Process. Retrieved October 27, 2017, from https://splendordesign.com/business-insight/mobile-app-ideation/

Statista. (2017a). App stores: number of apps in leading app stores 2017. Retrieved November 28, 2017, from https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

Statista. (2017b). Google Play Store: number of apps 2009-2017 | Statistic. Retrieved November 28, 2017, from https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/

Statista. (2017c). Mobile app revenues 2015-2020 | Statistic. Retrieved November 28, 2017, from https://www.statista.com/statistics/269025/worldwide-mobile-app-revenue-forecast/

Sun, Z., Ji, Z., Zhang, P., Chen, C., Qian, X., Du, X., & Wan, Q. (2017). Automatic labeling of mobile apps by the type of psychological needs they satisfy. *Telematics and Informatics*, *34*(5), 767–778.

Vakulenko, S., Müller, O., & Brocke, J. (2014). Enriching iTunes App Store Categories via Topic Modeling. Presented at the International Conference on Information Systems, Auckland, New Zealand. Retrieved from http://aisel.aisnet.org/cgi/viewcontent.cgi?article=1345&context=icis2014

Wang, A., An, N., Lu, X., Chen, H., Li, C., & Levkoff, S. (2014). A Classification Scheme for Analyzing Mobile Apps Used to Prevent and Manage Disease in Late Life. *JMIR MHealth and UHealth*, *2*(1).

Woerndl, W., Schueller, C., & Wojtech, R. (2007). A Hybrid Recommender System for Context-aware Recommendations of Mobile Applications. In *2007 IEEE 23rd International Conference on Data Engineering Workshop* (pp. 871–878). IEEE.

Zakaryazad, A., & Duman, E. (2016). A profit-driven Artificial Neural Network (ANN) with applications to fraud detection and direct marketing. *Neurocomputing*, *175*, 121–131.

Zhou, J. (2016). *Automated App Categorization using API analysis*. University College London. Retrieved from http://www0.cs.ucl.ac.uk/staff/Yue.Jia/resources/studentprojects/Kelly_Zhou_Automated_App_Categorization_using%20API_analysis.pdf

Zhu, H., Chen, E., Xiong, H., Cao, H., & Tian, J. (2014). Mobile App Classification with Enriched Contextual Information. *IEEE Transactions on Mobile Computing*, *13*(7), 1550–1563.

Zhu, Xiong, H., Ge, Y., & Chen, E. (2013). Ranking Fraud Detection for Mobile Apps: A Holistic View. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management* (pp. 619–628). ACM.

Zimmermann, A., Lorenz, A., & Oppermann, R. (2007). An Operational Definition of Context. In *Proceedings of the 6th International and Interdisciplinary Conference on Modeling and Using Context* (pp. 558–571).

Appendices

A. Text Analysis

Table A- 1 : Entity Type Identified by SAS

	Entity type
1	Address
2	Company
3	Currency
4	Date
5	Internet
6	Location
7	Measure
8	Organization
9	Percent
10	Person
11	Phone
12	Product
13	Ambiguous proper noun
14	Social Security number
15	Time
16	Time period
17	Title
18	Vehicle

Table A- 2 : Descriptive Terms of Clusters

ClusterNo.	Descriptive terms	Frequency
1	wallpaper image hd picture live screen set wallpapers beautiful background	120
2	friend message whatsapp sms share social facebook love picture card	94
3	android support weather apps screen watch battery install phone version	457
4	find search deal price shop offer book store travel home	195
5	learn question practice test english word student score help tool	261
6	news late story article break read stay video offline live	120
7	wifi network vehicle connection vpn car internet connect service data	133
8	read book experience story world start enjoy daily health practice	402
9	share effect photo filter edit add create facebook picture image	399
10	guide game tip play trick game fan player cheat copyright	84
11	draw color drawing step style kid create look tool image	140
12	account mobile service card email manage credit balance message access	163
13	map gps route location track offline navigation place plan travel	239
14	track weight day daily exercise plan health step parent start	294
15	file document mode android support screen device note edit video	272
16	live news watch team score highlight tv game english player	200
17	kid child color play game fun parent fun toy learn	89

Table A- 3 : Clusters of Apps

Google category	App name	ClusterNo.	...	Cluster3	Cluster 4	...
Social	Messenger for Facebook	3	...	100%	0%	...
Communication	Calls Blacklist - Call Blocker	4	...	0%	100%	...
Communication	WeChat	4	...	0%	100%	...
Productivity	Contacts Optimizer	4	...	0%	98%	...
Dating	Wapo: Gay Dating	3	...	100%	0%	...
Communication	Burner - Free U.S. Number	4	...	0%	100%	...
Business	join.me - Simple Meetings	4	...	0%	100%	...
Social	Badoo - Free Chat & Dating App	3	...	100%	0%	...
Tools	Kika Keyboard - Emoji, GIFs	14	...	0%	0%	...
Tools	CoolSymbols emoticon emoji	14	...	0%	0%	...
Communication	Glide - Video Chat Messenger	4	...	0%	98%	...
Communication	ExDialer - Dialer & Contacts	4	...	0%	100%	...
Personalization	Glass GO Launcher Theme	14	...	0%	0%	...

Table A- 4 : Ranges of Topic Weights

Topic ID	Topic weight(Min)	Topic weight(Max)
1	0.094	0.614
2	0.109	0.600
3	0.096	0.791
4	0.100	0.660
5	0.084	0.627
6	0.071	0.374
7	0.093	0.553
8	0.083	0.471
9	0.079	0.411
10	0.078	0.561
11	0.075	0.492
12	0.094	0.358
13	0.090	0.816
14	0.068	0.286
15	0.098	0.464

B. Prediction Models

Figure B- 1 : Performances of Rule-Based Models

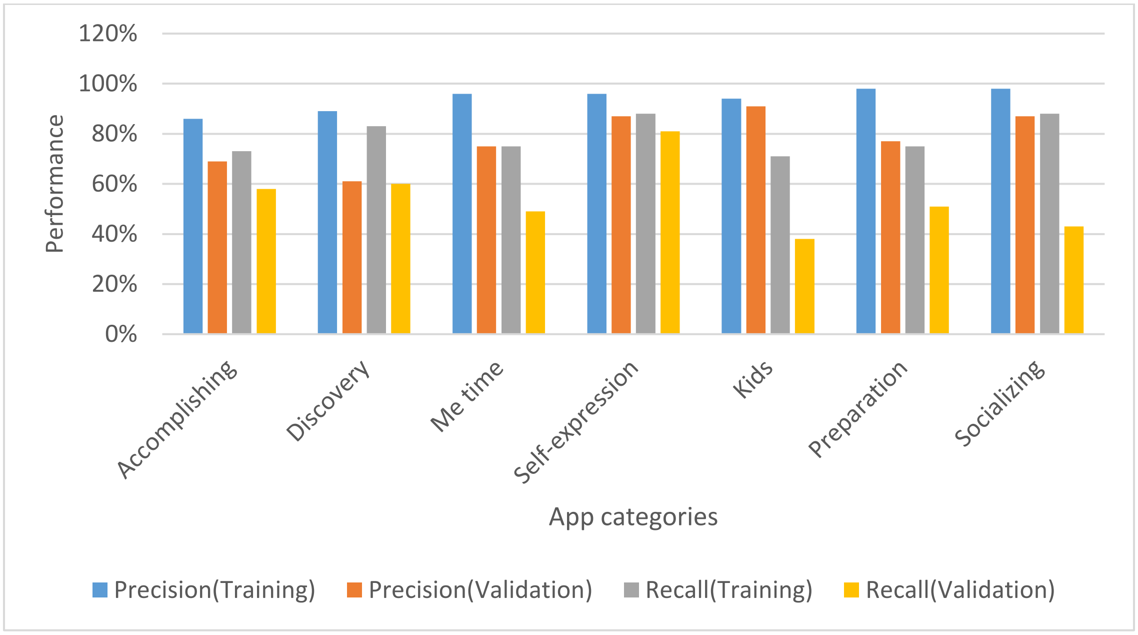


Figure B- 2 : Performance Errors of Decision Trees

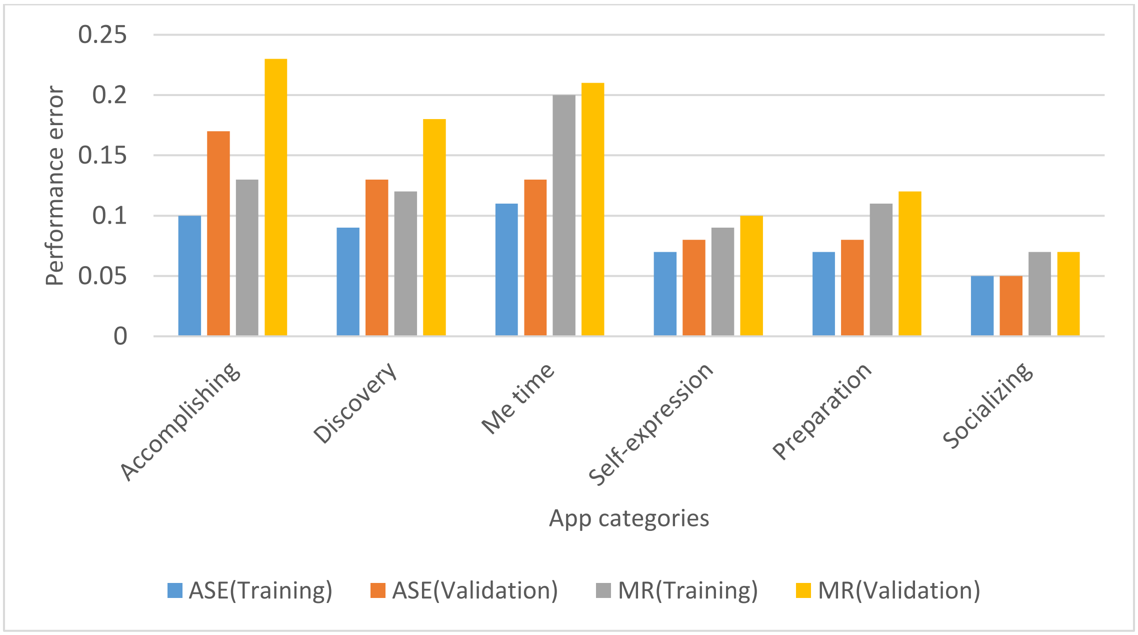
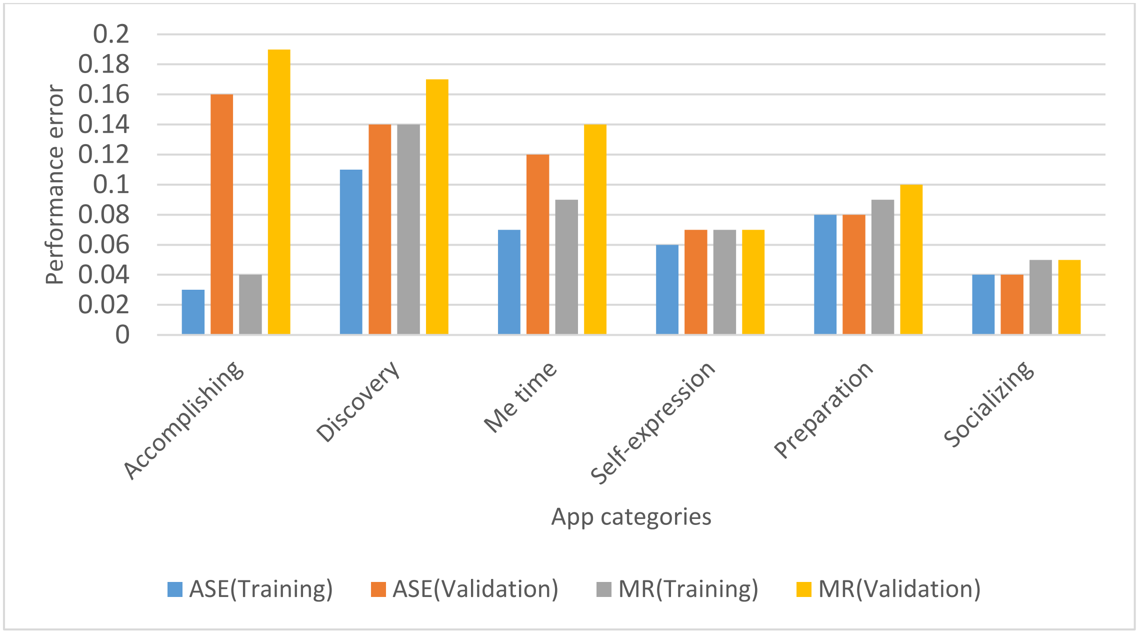


Figure B- 3 : Performance Errors of Artificial Neural Networks


Performance error
Accomplishing
Discovery
Me time
Self-expression
Preparation
Socializing
App categories
ASE(Training)
ASE(Validation)
MR(Training)
MR(Validation)