



INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

BACHELOR'S THESIS PROJECT

AUG - NOV 2013 JAN - APR 2014

---

## **Mobile Based Recommendation Systems, Sharing of Information while Ensuring Privacy**

---

*Authors:*

Amerineni Rohith (10010109)  
Revanth Bhattaram (10010153)  
B.Tech Final Year  
Department of Computer Science  
IIT Guwahati

*Supervisor:*

Dr. Sanasam Ranbir Singh  
Associate Professor  
Department of Computer Science  
IIT Guwahati

# Contents

<b>PREAMBLE</b>	<b>3</b>
<b>ABSTRACT</b>	<b>4</b>
<b>1 INTRODUCTION</b>	<b>5</b>
<b>2 PROBLEM DEFINITION</b>	<b>6</b>
2.1 Challenges . . . . .	6
2.2 Background Studies . . . . .	7
2.2.1 Existing Frameworks . . . . .	7
2.2.2 How does our method stand out ? . . . . .	7
<b>3 PROPOSED FRAMEWORK</b>	<b>8</b>
3.1 Building a User Profile . . . . .	8
3.1.1 Collecting Information . . . . .	8
3.1.2 Classifying Activities . . . . .	8
3.2 Building the Information Sharing Network . . . . .	8
3.2.1 Framework Design . . . . .	8
3.2.2 Functionality . . . . .	9
3.3 Learning from partial information . . . . .	10
<b>4 PERSONALIZATION ANALYSIS</b>	<b>10</b>
4.1 Personalization through Browsing History . . . . .	10
4.2 Personalization through Participation History . . . . .	11
<b>5 PRESERVING PRIVACY</b>	<b>13</b>
<b>6 DEPLOYMENT IN MOBILE</b>	<b>14</b>
6.1 Collecting and Classifying . . . . .	14
6.2 Network . . . . .	14
6.3 Database . . . . .	15
6.4 Working of the App . . . . .	16
<b>7 CONCLUSION AND FUTURE WORKS</b>	<b>17</b>
<b>Appendices</b>	<b>18</b>
<b>A Classifier</b>	<b>18</b>
A.1 Building the Classifier . . . . .	18
A.1.1 Class Levels . . . . .	18
A.1.2 Feature Extraction . . . . .	18
A.1.3 Naive-Bayes Classifier . . . . .	18
A.2 Analysis Of Classifier Performance . . . . .	19
A.3 Questionnaire for Survey . . . . .	20

## **PREAMBLE**

### **Timeline of Work Done:**

The following is an account of the work done in the two phases:

#### **Phase I:**

- Exploring the extent to which mobile activity information can be collected.
- Establishing a set of class levels and the means by which the user profile is represented.
- Building an initial classifier to be used as a base for the recommendation system.
- Applying the classifier on mobile activity data and being able to classify activities to a class level.
- Implementing an iterative learning algorithm which provides the personalization of the system.
- Worked on making an efficient database design keeping in mind, the constraints of mobile devices.
- Deploying the profiler and classifier and testing them on devices. Performance issues were noted and were fixed using the required optimizations.

#### **Phase II:**

- Exploring different methods for personalization - through activities, participation etc.
- Designing the query system. Type of questions, feedback etc.
- Being able to provide recommendations given similarity scores and answers of various users.
- Analyzing various similarity measures and their accuracies with respect to the actual data.
- Establishing the information sharing network and taking care of security and performance issues.
- Deploying and testing the framework developed, on mobile devices.

### **Individual Contributions:**

The following are the individual contributions throughout the project. It must be noted how several of the work was shared and it is hard to pin point exact individual contributions.

#### **Amerineni Rohith:**

- Mobile Profiling - Collecting mobile activities such as phone log, messages, email.
- Development of the classifier and analyzing accuracy.
- Analysis of various similarity measures and accuracies.
- Front End of the Application

#### **Revanth Bhattaram:**

- Mobile Profiling - Collecting mobile activities such as browsing data, facebook likes.
- Design of the database schema.
- Design of the information sharing network and handling issues of privacy.
- Development of the Android App Back End and deploying the classifier and network components of the framework.

## **ABSTRACT**

The usage of mobiles and tablets are on par with laptops and desktops. This change is due to an increase in the computational power, the wide variety of apps available and the portability of these devices. The amount of personal information that is stored on a person's mobile device has increased enormously. Mining this information will allow us to construct a user profile which could then be used to provide recommendations that the user would find useful. Another application is the ability to find a set of similar users based on the constructed profiles. Grouping these users and sharing information amongst them would be of great use in small communities. This group of similar users can also be used a query system where a user can post a query and can get the opinion of other users who deemed to be similar. Another issue that is handled is the necessity to maintain a user's privacy. Unlike traditional recommendation systems and classifiers where a user's profile data is stored on a server, the proposed classifier does all the processing on the user's mobile device while using minimal resources. This ensures that the privacy of a user is not compromised.

# 1 INTRODUCTION

Thirty years ago, the term mobile phone would put in a mental picture of a fairly big box (Figure1) limited to setting up a connection between two people in our mind. Over the years, the sizes of these mobile devices have reduced considerably and have seen huge leaps in computational power. So much so that there are even 64 bit mobile phones out there that are being produced by Apple Inc<sup>®</sup>. People are so attached to these devices that they tend to be updated with latest models and latest software. With the support of memory cards, the amount of personal data stored by an individual in his mobile has also increased. Seeing all this we can undoubtedly state that mobile phones are becoming increasingly personalized in terms of the data they store and the types of services they provide (Karlson et al., 2009). These small portable devices with great hardware specifications and an abundance of application support have grabbed the attention of the present generation and have become almost ubiquitous.

Recommendation system based on a user's personal data is not a new field. Recommendation systems apply different data mining techniques on data and try to find the similarity among a huge collection of data items and typically produce a list of recommendations in one of two ways - through collaborative or content-based filtering (Chen and McLeod, 2005). Collaborative filtering approaches build a model from a user's past behavior as well as similar decisions made by other users and then use that model to predict activities that the user may have an interest in. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties (Melville and Sindhvani, 2010; Mooney and Roy, 2000). There are also some systems which combine these two and are called as hybrid recommendation systems. These recommendation systems are personalized to each of the users and upon sharing some of the aggregate information that this personal system has, similar users can be found and used to improve the quality of recommendations.

Integrating these kind of systems into mobiles is a fairly new research area. With the increasing amount of smart phones are having internet access an increasing number, many people have been drawn into this are of mobile recommendation systems. The main challenge to deploy a recommendation system in a mobile is the effective usage of the available resources. We must not overuse the computation power of the mobile leading to the death of the battery. Also, the development of personalized recommender systems in mobile is much more challenging than developing recommender systems from traditional domains due to the complexity of spatial data and intrinsic spatiotemporal relationships, the unclear roles of context-aware information, and the increasing availability of environment sensing capabilities. There has been some work with respect to mobile recommender system and one such application is suggesting potentially optimal routes that maximize the profit for the taxi drivers (Ge et al., 2010). For such an application we need to know the location details such as latitude and longitude. So we need some device that is always with the driver and mobile is the device which best suits our needs.

Having this system in a smart phone, it lets us predict different traits of the user. All these depends on the extent to which we can mine the mobile data. In Android<sup>®</sup> upon the user permissions we can access wide amount of data such as list of applications installed and its usage statistics, Browser History, Emails, Social Networking data, Call Logs and so on. Profiling all these let us know the periodic activities done by the user with his mobile, the type of news and newspaper he generally reads, his financial status, his interests and also his personality which is of utmost importance to any recommendation system

Next, having this recommendation system in many users' mobiles, they are personalized according to the respective usage. Hence, these recommendation systems are in a way the representation of the user's profile. Taking a further step, we can group the users who are having a common personalized recommendation systems. This can be done by sharing the aggregate information such as rating of the item of interest. Such information is useful when you want to judge about the interest of a particular item for a user - Ask all the similar users about their interest about this product and judge the interest of this user based on the information that is shared by others which is called collaborative learning (Melville and Sindhvani, 2010). So sharing information in such a way is of great importance but the issue of privacy should be looked into and only the relevant and safe data must be shared to other users of this system.

The rest of the report is organized as follows. The second sections presents you with the problem statement and the work split in the two phases. The third section elaborates the process undergone to come up with the basic recommendation system to start with and the progress of the profiling done in mobiles. It also explains the progress of the deployment of the recommendation system in mobile and its integration with the other profiling



Figure 1: Mobiles in their initial stages. They were very big and its sole purpose is only talking to other people having this device

data resources. Fourth section presents you the results we got from the system developed until now and some design prototypes followed in the mobile.

## 2 PROBLEM DEFINITION

Having this introduction with the area we are dealing with, our target in brief is to develop a system which servers the following purposes

1. Build a mobile recommender system using minimal phone resources that is it should take less computation power
2. Improve this recommendation system and personalize it specific to the user by iterative learning
3. Group the user and provide inter system communication for better performance and protect user privacy

The work of action in order to obtain this is presented in the forthcoming sections.

### 2.1 Challenges

- The first part of the project is to build a classifier which is used to classify each activity of the mobile. We have to create some class levels and identify some features to build any classifier. The initial classifier built will not be specific to the user and this problem of cold start is present in any recommender system.
- Since we are using this system in mobile, we cannot keep many classes and features as it requires high computation. So we try to minimize the class levels and feature set which affects the efficiency of the classifier. Even after decreasing the classes and features by a considerable amount, we still have a database file of 24MB to be stored in the mobile.
- The dataset we used for building the classifier is taken from <http://www.dmoz.org>. These web pages have only the meta-data and one line description of the web pages. Hence getting the features from this sparse data was not very accurate. Although this data represents the document exactly it might not have any word from the feature set we selected. Hence this document is not classified by our classifier leading to low values of precision and recall.
- Unlike the traditional recommender systems which rely upon the browser history we need to gather other data as well. Since in mobiles many people tend to use applications for any famous website and regular activities like news reading. Hence if we completely depend upon the browser history, we are sure to miss a lot of data. But accessing the information of the application is not always feasible and legal.
- For the inter mobile communications that we plan for grouping users and sharing the information we need to set up the connection between the mobiles. For this we cannot access the IP of the mobile phone as it is not consistent. So we need a server and this would require continuous monitoring and security of data needs to be ensured. We need to look as to how to set up a peer to peer connection.

## 2.2 Background Studies

This section gives an overview of the various types of existing recommendation systems and aims to show how the idea proposed in this paper stands out.

### 2.2.1 Existing Frameworks

Recommendation systems are a class of techniques that are designed to predict user responses to options/opinions.

Two very popular approaches used to design recommendation systems are:

#### 1. Collaborative Filtering Chen and McLeod (2005):

A famous example where collaborative filtering is used is Amazon.com's product recommendation system. Large datasets of transactions are analyzed and items which are usually bought together by many users are found. This is how the "people who bought X also bought Y" suggestions are made on the website.

Media websites like Netflix use collaborative filtering to recommend content based on what similar other(similar) users have gone for.

Social networks is another area where collaborative filtering finds use. By making observations on the user's social network, friend, groups and page recommendations are made.

#### 2. Content Based Filtering:

IMDB uses content based filtering to provide movie recommendations. A movie is recommended to a user by measuring the similarity between the description and the descriptions of movies which have been seen/liked by the user.

Amazon's book recommendation system works in a similar manner. By comparing the contents of two books, a similarity measure is established and suggestions are made.

An observation here is that most systems are web based where the user's activities are monitored and stored in an online data store.

### 2.2.2 How does our method stand out ?

The following are the aspects in which our method stands out:

- **Mobile based:** As mentioned above, most recommendation systems are web based and use information obtained from the time a user spends on their website/application. The proposed system, on the other hand, analyzes the data present on a user's mobile. In times where the usage of mobile phones is increasing exponentially, the amount of personal information present on a mobile device is quite substantial. Also, unlike most recommendations systems that collect data pertaining to a single domain/area, analyzing mobile data will help in constructing a more *global* profile of a user.
- **User data:** The design of the system is such that the user's activity data is stored and processed only on the mobile device. This makes it virtually impossible for the user's personal data to leave the device. It is invulnerable against *attacks* that plague traditional systems.
- **Privacy:** The user's identity and his personal details are kept hidden at all times and are not shared with anyone else. Compare this with the above mentioned recommendation systems where the identity of a person might be revealed while providing recommendations.
- **Computation:** Most of the computation takes place on the user's device itself instead of the usual choice of an external server. The computation has been designed to be lightweight to work on any contemporary mobile device without requiring excessive computation power. The only part of the system that doesn't function on the device itself is collecting opinions from similar users and providing recommendations

## 3 PROPOSED FRAMEWORK

### 3.1 Building a User Profile

#### 3.1.1 Collecting Information

To build any recommender system, the primary thing we need is the data about the user. The more we have the data about the user, the more the system works. With respect to this, privacy is the main issue which is of concern. The traditional recommender system uses the browser history for recommending some ads on your browser as told in the previous section. But in mobiles, we have much data other than browser history such as applications installed, social network data, calendar activity and so on. As a first step we looked into all the data that we could gather in the mobile upon user permissions. We have built an Android application that gathers the following data

1. Browser History
2. Facebook Likes, Emails
3. Applications Installed
4. Call Logs

Apart from this data we plan to get the data pertaining to the periodic activities which are done by the user in his mobile. All this data is only used on the client side and nothing is exported to the server. As our prime motive is to classify every mobile activity we had gathered all the possible data and not just restricted to browser history. However to start with, we are using browser history to ensure the proper working of the system.

#### 3.1.2 Classifying Activities

Using the classifier described in the Appendix A, the activities collected in the previous step are analyzed and are assigned a class level. An iterative learning mechanism has been put in place which is used in personalizing the classifier.

As more and more activities are collected, the extent of personalization increases. In addition, with more usage, we will be able to get a more accurate profile of the user which will be essential in getting recommendations and comparing similarities between users.

It must be noted how the building of a user profile is intended to take place completely on the mobile device. This is one of the main principles of the mobile based recommendation system.

### 3.2 Building the Information Sharing Network

To allow users to gather and share opinions, a system is to be developed. For this project, a network framework that lets users ask for feedback and share their own opinions has been designed. The designed system minimizes load on the user end while ensuring privacy.

#### 3.2.1 Framework Design

The network is built on a peer-to-peer (P2P) architecture using the SIP implementation. The nodes in the network are of two types:

- **Bootstrap Node:** This serves as the initial point of contact for newly joining nodes. The bootstrap node maintains a list of user nodes in the network along with a response repository.
- **User Node:** All user mobile devices that connect to the network fall under this category. User nodes can interact with any other node in the network.

The user can get feedback by sending a message through the network. In addition, the user can also sync data from the response repository thereby ensuring that the user will be able to obtain recommendations right out of the box.

The advantage of using such a framework is how one will be able to receive responses almost instantly from the other users present in the network. It must be noted that for the user to receive responses, they have to be connected to the network at all times.



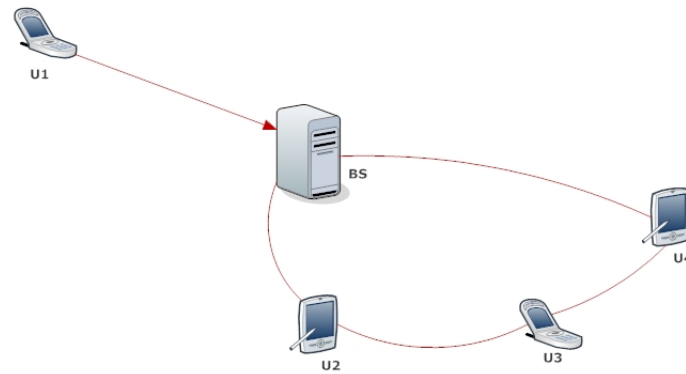


Figure 2: Connection Network Diagram

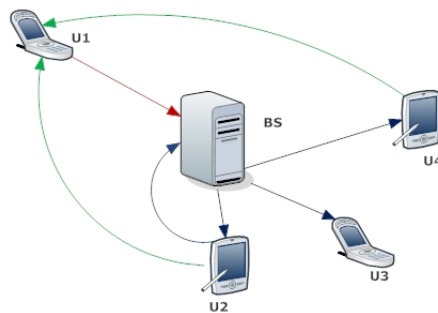


Figure 3: Feedback Network Diagram

### 3.2.2 Functionality

The following are the functions available for a user in the network:

1. **Connecting to the network:** To join the P2P network, the user node sends a join request message to the bootstrap node. The bootstrap node, which maintains a list of connected nodes at all times, adds the user node to the list of connected nodes and thus to the network.

Fig.2 is the network diagram for this activity. The user node (U1) sends a join request message to the bootstrap node (BS) which already has the other user nodes (U2,U3,U4) in the network.

2. **Getting feedback:** When a user wishes to get feedback, the question, along with their profile is sent to the bootstrap node which broadcasts the message. When someone decides to reply, their reply, along with the computed similarity score is sent directly to the asker.

The responses that a user receives are stored on the device and the net feedback is found by performing a weighted average on the responses where the similarities serve as weights. If the answerer allows the response to go public, a message containing the question and the answer is sent to the bootstrap node which stores this data in its response repository.

Fig.3 shows the network diagram for this activity. User U1 wishes to get feedback for his question and so sends a message containing the question and his profile to the bootstrap node. The bootstrap node then sends the message to all other nodes in the network.

Now, User U3 does not choose to reply and so no messages are sent. User U2 replies and allows his reply to go public. As shown in the figure, two messages are sent from U2: one to U1 and another copy to the bootstrap node. User U4 replies but chooses to not make his reply public and so only one message, addressed to the asker, is sent.

3. **Sync Repository Data:** The user will have his own copy of the response repository albeit outdated when compared to the one maintained by the bootstrap node. The user can synchronize his copy with the main

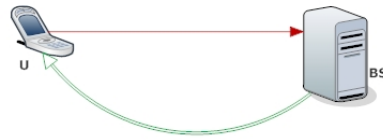


Figure 4: Syncing Network Diagram

repository by sending the bootstrap node a 'Repo Update' request. Information about the last response stored in the local repository is sent so that only the required responses are sent by the bootstrap node.

Fig.4 shows the network diagram of this activity. The user U, sends a message containing the request and a 'Max Response Id' to the bootstrap node. In return, the bootstrap node sends all responses that are 'missing' from the user's local copy of the repository.

### 3.3 Learning from partial information

The information obtained the response repository is not an entire log of all the responses that take place in the network. If a user wishes to make their feedback private, then that response is not recorded for everyone but is available only to the one who asked the question. So, the response repository consists of responses that have been authorized for public distribution and the local repository of each user differs and have their own partial data. With this table we can get the behavior of the users in a particular category by average rating, class entropy, patterns in his rating based on some keyword in that category whose analysis will be given later in the following sections.

## 4 PERSONALIZATION ANALYSIS

The aim of personalization in the mobile is planned based on the following three directions.

### 4.1 Personalization through Browsing History

This is the learning part of the application. To test the personalization aspect using browsing history, two different user profiles were used and the list of features was recomputed.

- **User - 1:** This user is a fan of the popular Japanese anime, Dragonball. Out of the 21 unique websites visited by the user, most are :
  - Wikipedia articles on popular Dragonball Z characters.
  - Fan sites and forums dedicated to the anime and manga.
  - Web pages related to video games based off the anime series.

After recomputing the features list, the following changes are made to the base features list. Features in bold are highly discriminating terms and are great features for classification

- **Features Removed** : absorpt, arid, bulli, childhood, cho, confin, decent, didn, dracula, fuji, grandmoth, grin, insert, invad, jelli, mutant, peril, recur, regen, roam, shy, slice, surprisingli, tend, verifi, watch.
- **Features Added** : afraid, **boo**, burst, dub, epoch, fierc, furri, **gohan**, impli, jin, **krillin**, lookout, miki, rivalri, stronger, summon, surpass, **teleport**, useless, wrath.
- **User - 2:** This user is a cricket fan. In the wake of Sachin Tendulkar's retirement, several news articles found on Cricinfo as well as other mainstream news sources are accessed. After recomputing the features list, the following changes are made to the base features list. Features in bold are highly discriminating terms and are great features for classification.
  - **Features Removed** : alleg, bulli, characteris, childhood, crouch, decent, decis, deploi, loss, mat, mob, mutant, rebuild, record, saga, sebastian, surprisingli, task, trunk, unnecessari, watch.


























				
				
				
				
				
				

Figure 5: Collaborative Filtering - This image shows an example of predicting of the user's rating using collaborative filtering. At first, people rate different items (like videos, images, games). After that, the system is making predictions about user's rating for an item, which the user hasn't rated yet. These predictions are built upon the existing ratings of other users, who have similar ratings with the active user. For instance, in our case the system has made a prediction, that the active user won't like the video. [Source:Wiki]

- **Features Added** : aunt, bangladesh, bharat, boyc, broke, chatter, cheer, couldn, **cowan**, crucial, exquisit, farewell, forget, gentleman, goodbye, humbl, hyderabad, inevit, kolkata, lanka, longev, maharashtra, **masterclass**, mere, **pavilion**, pradesh, punjab, rever, roar, **sachin**, sad, salut, spoke, steen, stood, struck, stump, surfer, **sweep**, tear, **tendulkar**, terrif, thrive, till, void, youngest.

This way all the links in the browsing history are crawled and classified into the respective classes using the above classifier. Now we will have updated features and the class distributions are updated for the user. These updated class distribution of the user forms a vector which is used to find similarity between the users.

## 4.2 Personalization through Participation History

To get the effective results from the Question and Answer interface that is provided in the application we need a large number of users to actively participate and use this application which would be a difficult task at this stage. So we simulated this type of participation by creating a google form and took a survey containing of about 57 questions. The form can be found at <http://goo.gl/TIfNvB>. This survey was taken by 75 random users and were asked to give rating on a scale of 1 to 10 where the users need not give rating to each and every question. Now we have the matrix which contains the rating of every user to some of the questions which can be seen as in (Figure5). These ratings form the characteristic vector of the user which can be used to find similar users based on cosine similarity measure. Now we can guess the ratings for the questions which the user did not give the his rating using the collaborative filtering technique.

This survey can be seen as a participation in the application as the only difference is that in the application the user can add his own questions but here we have fixed the questions. Also in the application, the user can make his response to the question as a

- Public Response - The rating given by the user is visible to all the users in the community
- Private Response - The rating given by the user is only visible to the one who is the owner of this question.

This privacy aspect is not covered in the survey part where all the responses are applicable to everyone. This public and private response in the application will lead to difference of similarities between the users and also preserves the privacy.

Having the opinions from a significant number of users we can now look at the traits of the user. For this, first we classified all the questions manually into the following categories - Politics, Movies, Cricket, Animation, TV Shows, Entertainment, Technology, News, Social, Football, Sports, Tennis. After having the basic categories in which we asked the questions we now see the variation of users' behavior in these categories and guess the rating for which the user has not provided himself. The recommendation score of a user  $u_i$  against a query  $q$  where SIM

Table 1: Characteristic Vectors For Average Rating

Category	U1	U2	U3	U4
Politics	6.80	5.16	4.33	5.00
Movies	6.18	6.38	5.50	6.50
Cricket	6.50	6.91	5.25	6.75
Animation	8.66	5.66	6.00	2.00
TV Shows	7.2	8.20	7.50	8.50
Entertainment	7.44	8.10	6.66	6.71
Technology	5.40	6.20	4.40	2.50
News	1.00	1.00	4.00	6.00
Social	6.16	6.80	4.57	4.28
Football	7.33	4.16	3.00	6.00
Sports	5.00	1.00	3.00	5.00
Tennis	5.00	5.00	4.50	7.50

Table 2: Similarity Between Users For Average Rating

Users	U1	U2	U3	U4
U1	1.0000	0.9575	0.9581	0.8962
U2	0.9575	1.0000	0.9679	0.8944
U3	0.9581	0.9679	1.0000	0.9399
U4	0.8962	0.8944	0.9399	1.0000

is the sum of all the similarities of  $u_i$  against others

$$score(q, u_i) = \sum_{u_j \in \mathcal{U}} sim(u_i, u_j).reco(u_j, q)/SIM$$

Calculating the similarity is of main concern here. The following describes some of the methods

**Simple Counting Method:** In this method, there is no calculation of similarity. We directly assign the average rating for that question given by the rest of the users as a recommendation to this user.

**Similarity Based on Average Rating:** In this method, we construct the characteristic vector of the user whose number of dimensions is same as the number of unique classes into which the questions were classified which are listed above. The value in each category is the average rating given by the user in that category. This way vectors are constructed for every user and these vectors are shown in the table 1 for some four random users. Now the similarity can be constructed between these vectors using cosine similarity measure. In this method, we can see that the average rating in a particular category is not varying much among the users and hence that would not be a good measure to find similar users. This can be observed from the table 2 as all the similarity values are high.

**Similarity Based on Class Entropy:** The characteristic vector of the user did not give us skewed and varied distributions among the users by the above method. We want to get another measure in that direction. One such method is by taking the entropy of each user in the each category and use them to construct the vector. See the entropy of the user's rating to all the questions in a category and get the vectors as in table 3. Here we see that the vector varies from user to user and the similarity values between any two users are not uniform as in table 4.

This choice of constructing the vector gives better description of the user and this can be deployed in the application as the data does not differ in the mobile.

Having this similarity, we want to get the score of the users to some questions for which he has given the rating. For this, we selected randomly the following 10 questions and tried to guess the responses of the user to them.

1. Do you agree that chennai express is worth its collections
2. Do you like Comedy Nights with Kapil

Table 3: Characteristic Vectors For Class Entropy

Category	U1	U2	U3	U4
Politics	0.1514	0.1367	0.0485	0.0586
Movies	0.1598	0.1223	0.0662	0.0607
Cricket	0.0873	0.0682	0.0886	0.0525
Animation	0.0072	0.2062	0.0120	0.1851
TV Shows	0.0765	0.0159	0.0612	0.1851
Entertainment	0.0833	0.0109	0.0295	0.0686
Technology	0.1420	0.1521	0.1177	0.0072
News	0.09259	0.0925	0.3703	0.5555
Social	0.1252	0.1153	0.1194	0.0876
Football	0.0414	0.0834	0.0000	0.0780
Sports	0.4629	0.0925	0.2777	0.4629
Tennis	0.4629	0.4629	0.0040	0.0024

3. How much do you like comedy films
4. Do you support Narendra Modi
5. How much do you follow the news of Times of India
6. How much do you like Test Matches
7. Rate Mahendra Singh Dhoni as a captain
8. Are social networks furthering human contact
9. Is Facebook better than WhatsApp
10. Do you like fun music videos in a language you cannot understand (like Gangam style)

For these questions we guessed the ratings and calculated the standard deviation of the guessed ratings from the actual ratings in three directions where

$$StandardDeviation(u_i) = \sum_{u_j \in \mathcal{U}} (ActualRating - GuessedRating)^2 / 10$$

- **Browsing History:** Using web history as a user representor, we calculated similarity and came with the recommendation score based on the top 5 similar users for the above questions. For the users whose browsing history was available to us we had done this and got a standard deviation of **5.24, 0.80, 15.36, 8.20, 8.09, 8.13, 4.98** for each of the user and a average error of **2.13** per question per user.
- **Average Rating:** Using web history as a user representor, we calculated similarity and came with the recommendation score based on the top 5 similar users for the above questions. For the same users as above we had done this and got a standard deviation of **5.68, 0.99, 14.47, 7.89, 8.65, 5.94, 5.18** for each of the user and a average error of **2.11** per question per user.
- **Class Entropy:** Using web history as a user representor, we calculated similarity and came with the recommendation score based on the top 5 similar users for the above questions. For the same users as above we had done this and got a standard deviation of **5.30, 0.83, 14.66, 6.38, 7.67, 5.82, 5.58** for each of the user and a average error of **2.07** per question per user.

The above values suggest that the method of using class entropy and average rating in a particular class level would give comparable results like the traditional browser history based recommendations and thus using the participation history will be of some use.

## 5 PRESERVING PRIVACY

The stand-out point of this project is how the privacy of the user is maintained at all times. It is imperative to keep the user's identity/personal information private and secure. To ensure that privacy is preserved, several steps have been taken.

Table 4: Similarity Between Users For Class Entropy

Users	U1	U2	U3	U4
U1	1.0000	0.8070	0.6078	0.5565
U2	0.8070	1.0000	0.3868	0.3753
U3	0.6078	0.3868	1.0000	0.9263
U4	0.5565	0.3753	0.9263	1.0000

- The process of collecting activity data and classifying takes place entirely on the user's mobile device. This makes it impossible for any information to leave the user's grasp.
- The identity of the user is protected by encrypting the userId before connecting to the network. When the user connects to the network, the hash digest of the user's device is used as a key to connect to the network. This makes it impossible for anyone to decrypt and obtain the user's identity.
- The bootstrap node is the only node in the network with information about other nodes . The user nodes in the network have no information/means to interact directly with other nodes.
- The only piece of information that is shared is a user's profile which is essentially a distribution of class levels. This information is never made known to the users through accessible interfaces. It is possible though, for packet sniffers to intercept network packets and pick up this data. However, to be able to reverse engineer and obtain a list of activities of the user is highly difficult.
- When a user answers a question, the only thing sent to the asker is the answer and the computed similarity score. The profile of the answerer is never shared. This is a step taken to protect a user's privacy.
- An important flexibility offered to the user is that they are in control of everything that flows in and out of their device. The user is not coerced to do anything. They could choose to not give any feedback at all, give private feedback or give feedback and make it public.
- Each user has tables similar to Table 1 which might make it possible for them get the profile of someone else. However, by encrypting user identities, we render any such effort useless.

## 6 DEPLOYMENT IN MOBILE

The devices used for the project run on the Android framework. The ubiquity of the Android operating system was the main reason why it was chosen over other mobile frameworks.

An app was designed, deployed and tested on an Aakash Ubislate tablet running Android 4.04 'Ice Cream Sandwich'.

### 6.1 Collecting and Classifying

Most of the work required in collecting data is done using standard android libraries. The activity data collected is stored in a database and can be used at any point in the future as well.

Once the user's activity data is collected, the above mentioned classifier is used to find out which class they fall into. Activities are classified and the respective classes of these activities are stored based on which the classifier is personalized (by recomputing features).

It must be noted how most of these functions take time to run. They have been designed to run in the background to ensure that the user doesn't face any performance issues.

### 6.2 Network

To implement the network framework, the sip2peer library was used. sip2peer is an open-source SIP-based middle-ware for the implementation of distributed and peer-to-peer applications or overlay, without constraints on device nature (fixed/mobile) and specific architecture.

The bootstrap node is deployed onto a central server, preferably with decent computation power. Interaction between peers takes place in the form of SIP messages. Different message types were defined and were used to perform the various required functions.

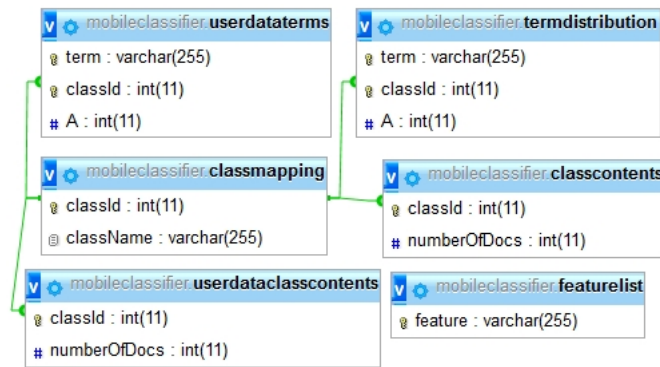


Figure 6: Classifier DB Schema

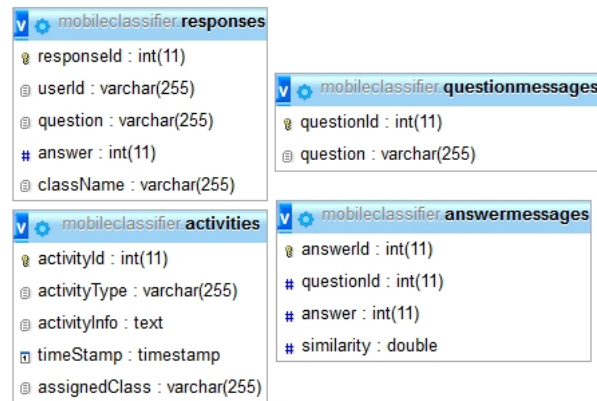


Figure 7: Mobile Activities DB Schema

### 6.3 Database

Now, since we wish to run the classifier completely on the user's mobile, we have to find a way to store all this information (activities, features) on the device itself. This screams for the usage of an in-memory database. We ended up using an sqlite relational database which is stored in the user's SD card.

Fig.6 and Fig.7 show the database schema. There are a total of five tables:

- **Class Mapping** : A mapping between class ID and the name of the class.
- **Class Contents** : This table serves as a mapping between class ID and the number of documents (encountered in the training data set) that come under this particular class.
- **Term Distribution** : This relation holds information about each term that has been encountered in the training data set. The A statistic (previously defined in the calculation of  $\chi^2$ ) is stored along with the respective class ID.
- **User Data Class Contents** : Similar to the class contents table except for the fact that this table is used to maintain exclusive count of user activities processed till now and the class they've fallen into.
- **User Data Term Distribution** : This is similar to the term distribution table except that this table is meant to store data related to the user activity data that has been processed.
- **Feature List** : This table consists of the list of features that are used while classifying document. Typically, these are terms that have high Gini coefficient values (  $> 0.95$  ) and are considered to be discriminating. This table is often recomputed so as to tweak the classifier to suit the user's profile.
- **Activities** : This table is used to store information about the various activities recorded on the mobile device. Activity Type tells you the type of activity (Web, Email, Call) and Activity Info contains data that will be

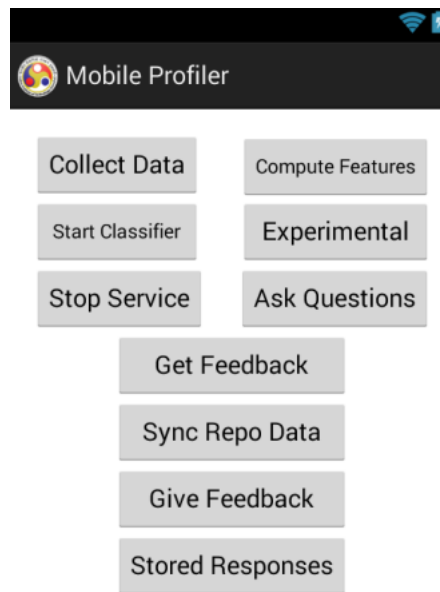


Figure 8: Basic App UI Interface.

used by the classifier to determine class to which this activity belongs to. Note that the timestamp of each activity is stored as well to ensure that activities are recorded only once.

- **Question Messages** : This table is essentially a list of questions that have been asked by the user.
- **Answer Messages** : This table contains the various answers (along with the similarity score) received for the user's various questions (ept track of by the questionId field).
- **Responses** : This is the user's local copy of the response repository stored with the bootstrap node.

In addition, the bootstrap node contains a database with the responses table present in it. All user devices sync data with this central repository.

The main challenge in designing the mobile app was to ensure that the app would not consume too many resources and at the same time would finish its tasks fairly quickly. The main bottleneck here is the database which is stored as a file in the SD card. For each query you would be performing I/O operations which are expensive, especially on a mobile device where the disk data transfer rates are not as high as their desktop counterparts. Therefore, the key step to optimizing performance was to reduce the number of I/O operations performed for each database operation. The queries that are used to interact with the database were written so as to ensure that minimum number of file I/O operations are performed.

## 6.4 Working of the App

The following are the various functions of the App. A basic GUI is shown in Fig.8

1. **Collect Data**: The user's mobile is scanned and relevant activity information (email, browsing history etc..) is collected and stored.
2. **Start Classifier**: The activity info captured in step 1 is fed to the classifier which then determines the class to which an activity belongs. This assigned class is recorded and the terms found in the activity document are used to update the classifier's term distribution.
3. **Compute Features**: This is the learning part of the application. Once several activities have been classified, the list of features are recomputed thereby personalizing the classifier.
4. **Ask Questions**: This redirects to an interface where the user can ask a question and send it to the network.
5. **Get Feedback**: This option lets the user view the net response to the questions that have been asked by them.



6. **Give Feedback:** This lets the user view questions asked by others in the network and answer them if it is desired.
7. **Sync Repo Data:** This sends out a 'repo update' message to bootstrap node that then replies with the required data, which is then inserted into the internal database.
8. **Stored Responses:** This is a basic interface which the user can use to view the contents of the local copy of the repository.

## 7 CONCLUSION AND FUTURE WORKS

In this project we have been able to construct a basic and efficient system that collects the various user activities found on a user's mobile device. Methods for network communication have been implemented to obtain feedback and interact with other peers. Multiple techniques of personalization have been analyzed and studied. The work done here serves as a foundation for future work.

In terms of future developments of this system:

- Activities other than the user's browsing history like emails, calendar activity, usage of apps, facebook data can be used for personalization as well. Another dimension to profiling is the ability to detect periodic activities in a user's activity log.
- The network portion of the project can be extended to provide support to more variety of questions/feedback type. Various personalization methods can be studied and hybrid methods can be thought of. In terms of privacy, data encryption during transmission is one of the areas of development for the project.
- In terms of recommendations, a potential area for future work is determining the reliability of a user. For example, a user who is always biased towards a particular topic/sub-topic might not give accurate/unbiased opinions. By going through the participation history of a user, it should be possible to determine the reliability of their feedback.

# Appendices

## A Classifier

### A.1 Building the Classifier

#### A.1.1 Class Levels

We had collected around 7.5L web pages which are classified into some predefined class heirarchy which can be found at <http://www.dmoz.org>. These web pages are not the entire HTML pages but just contain meta data and one line description. As a first step into building the classifier, we have to limit the number of classes as we are going to deploy this in a mobile. So we fixed the depth of class heirarchy to three and combined the classes of higher depth into this. Upon fixing the depth also we are left with a considerable number of classes. As a final step, we set a threshold that each class should have a minimum support of 1000 web pages in it. This reduced the number of class levels to 229.

#### A.1.2 Feature Extraction

After the class levels are decided we divided the entire data set into training and testing data (80:20). Next Step is the feature extraction for these class levels. There are a lot of of feature selection methods in the problem of text categorization namely document frequency (DF), information gain (IG), mutual information (MI), chi square ( $\chi^2$ ), term strength (TS) of which IG and  $\chi^2$  are found to be most effective (Yang and Pedersen, 1997). We used  $\chi^2$  statistic as a measure for feature selection.  $\chi^2$  is given by :

$$\chi^2(t, C) = \frac{N * (AD - CB)^2}{(A + C) * (B + D) * (A + B) * (C + D)}$$

where where N is the number of documents, A is the number of documents of class c containing the term t, B is the number of documents of other class (not c) containing t, C is the number of documents of class c not containing the term t and D is the number of documents of other class not containing t.  $\chi^2$  measures the lack of independence between t and C.

Having calculated the  $\chi^2$  values, we need to get the global goodness of the term. There are many methods which give the global goodness estimation functions such as mean and maximum. Using these functions are not guaranteed to give the features which describes the class most. A feature describes the class most when it is present only in a particular class. A feature which is uniformly distributed over all the classes are not good features for a classification purpose. So, the global goodness of the features can be identified by looking at the features distribution among the classes. But, the above mentioned mean and maximum do not capture this. Hence, we have used Gini Index to calculate the global goodness of the terms. Gini Index ( also known as Gini Coefficient or Gini Ratio ) is used to measure the inequality among the values of the distribution. It is defined based on the Lorentz curve as in Figure9 which plots the proportion of the total income of the population (y axis) that is cumulatively earned by the bottom x% of the population (Gastwirth, 1972). If the value of the Gini Index is 0, then it can be seen from the Figure9 that the distribution is uniform. Likewise, a higher value of the stat implies skewed/unequal distribution. Using this statistic and setting a threshold of 0.95, we finally arrived at the set of features to build the classifier.

#### A.1.3 Naive-Bayes Classifier

A classifier is a function f that maps input feature vectors  $x \in X$  to output class labels  $y \in 1,2,3,...,C$  (Murphy, 2006). A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with independence assumptions. On a simple note, it assumes that the presence or absence of a particular feature is independent of the presence/absence of any other feature, given the class level. For example, an Indian can have a lot of features, such as height, weight, complexion and so on. All these features are independent of each other and each of them contribute a certain probability that a person with that feature is an Indian.

On a mathematical side, as stated earlier, it is based on Bayes' Theorem. The classifier can be seen as, Given a set of features, we have to decide which class ( $C_i$ ) does these features ( $F_1, F_2, \dots, F_n$ ) direct us to i.e.

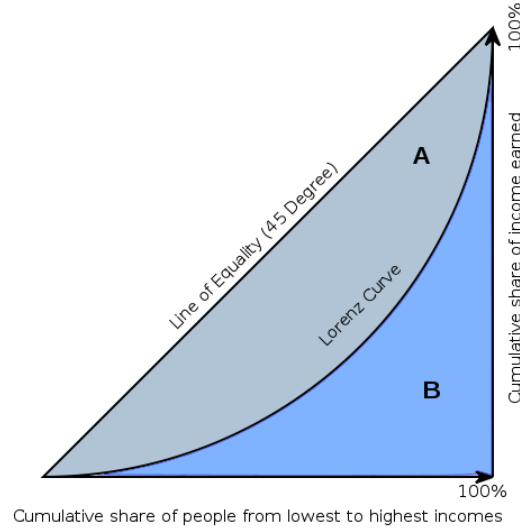


Figure 9: The graph shows that the Gini coefficient is equal to the area marked A divided by the sum of the areas marked A and B. that is,  $Gini = A / (A + B)$ . It is also equal to  $2*A$  due to the fact that  $A + B = 0.5$  (since the axes scale from 0 to 1). Source: Wiki

$P(C_i | F_1, F_2, \dots, F_n) > P(C_j | F_1, F_2, \dots, F_n)$  where  $i \neq j$  and  $(i, j) \in (0, \text{No. of Classes})$ . Using Bayes' theorem, this probability can be written as :

$$P(C_i | F_1, F_2, \dots, F_n) = \frac{P(F_1, F_2, \dots, F_n | C_i)P(C_i)}{P(F_1, F_2, \dots, F_n)}$$

In plain english, the above equation can be seen as :

$$posterior = \frac{prior * likelihood}{evidence}$$

As we can see, the denominator of the RHS is constant for a feature set over all the classes and hence can be ignored for the purpose of calculations in the practical scenario. Using the chain rule, we finally end up with the below equation

$$P(C_i | F_1, F_2, \dots, F_n) = P(C_i)P(F_1 | C_i)P(F_2 | C_i) \dots P(F_n | C_i)$$

For these classification purposes we need to calculate all the probabilities of a feature with all the class levels and need to be stored. Once these values are calculated from the dataset, when a new document arrives we can apply the above model to classify it.

## A.2 Analysis Of Classifier Performance

The classifier that is initially built is tested on a testing data set. It classified 71845 correctly which is 45.53% of the dataset and 75649 documents were assigned the wrong class level. Apart from these, 10299 of the documents were not assigned any class level, because there were no features in the documents. Upon close examination of the results it is found that, the class levels of the documents which are incorrectly classified are closely related to the actual class level. For example, a document in Anime is classified into cartoon class level. This inaccuracy is because of the short length of the documents and some of the class levels have the same set of features. As the final motive is to set up a recommender system and group the users, this number is good enough for a cold start. This is because, initially the same classifier is deployed everywhere. So when a text is wrongly classified for a particular user, then the same will be done to other users with a similar text. Hence, the way the class level distributions change will be similar and we get to know about the similar users. Although we need a good classifier to recommend things in a better manner, the way our classifier works is good enough to proceed forward with.

### A.3 Questionnaire for Survey

Below are the list of questions and their respective class levels which we used to construct the characteristic vector of the user as in table 3.

1. Do you support Aam Aadmi Party — Politics
2. How did you like Dhoom — Movies
3. Do you agree that chennai express is worth its collections — Movies
4. Did sachin take the right decision regarding his retirement — Cricket
5. How is the performance of India in the latest South Africa tour — Cricket
6. Do you think Jacques Kallis is a perfect all rounder of cricket — Cricket
7. Do you support Telangana Movement — Politics
8. Rate Iron Man series — Movies
9. Rate Tom and Jerry cartoon — Animation
10. Do you like WallE movie — Animation
11. Do you like animated movies — Animation
12. Rate prison break serial — TV Show; Movie; Entertainment
13. Do you think new Iphone is worth its cost — Technology
14. How much do you use apple products — Technology
15. Regional Film industries (Tollywood, Kollywood etc ) in India are comparable with the Bollywood — Movie
16. How scary was the movie Conjuring — Movie
17. Do you like Comedy Nights with Kapil — Movie
18. Do u like Comedy Circus — TV show; Movie; Entertainment
19. How much do you like comedy films — Movie
20. Do u oppose AAP taking the support of congress to form the government — Politics
21. Do you support Narendra Modi — Politics
22. How much do you follow the news of Times of India — News; Politics; Movies
23. How much do you like Test Matches — Cricket
24. Do you like the play of Rahul Dravid — Cricket
25. Do you like IPL — Cricket
26. Do you think IPL supports in identifying the local talent — Cricket
27. Rate Mahendra Singh Dhoni as a captain — Cricket
28. How much lucky do you think is mahendra singh dhoni — Cricket
29. Do you support film stars getting into politics — Movie; Politics
30. Do you agree that Quora is a good platform for getting knowledge — Social
31. Do you agree that boredom of facebook is the cause of rise of quora - Social
32. Do think that more than knowledge there a lot of trolls coming up in quora as in facebook — Social
33. Do you think video games are a good way to pass time — Technology; Entertainment
34. Are video games getting too violent these days — Entertainment
35. Are social networks furthering human contact — Social
36. Do you like football — Football
37. Would you prefer to play FIFA over a real life match of football — Football
38. Is The Dark Knight the best movie of all time — Movie

39. Do you prefer Football over Cricket — Cricket; Football; Sport
40. Is Quora here to replace Facebook — Social
41. Is YouTube better than Television networks — Social
42. Is Facebook better than WhatsApp — Social
43. Did you like the Gangam style video — Movie; Entertainment
44. Do you like fun music videos in a language you cannot understand (like Gangam style) — Movie; Entertainment
45. Is Nadal better than Federer — Tennis
46. Will Nadal beat Federers records — Tennis
47. Will Windows phone be able to do anything substantial — Technology
48. How do you like Windows 8 — Technology
49. Is SRK better than Amitabh Bachan - Movie
50. Is Mitchell Johnson the best bowler of the year — Cricket
51. Will Australia be able to regain its status — Cricket
52. Will Bayern Munich win the Champions league again — Football
53. Will Suarez win the EPL best player award — Football
54. Will Manchester United finish outside the top 4 this season — Football
55. Is Roadies overrated — TV Shows; Movie; Entertainment
56. Has MTV gone downhill — TV Shows; Movie; Entertainment
57. Are there too many reality shows going on these days — TV Shows; Movie; Entertainment

## REFERENCES

- Chen, A. Y.-A. and McLeod, D. (2005). Collaborative filtering for information recommendation systems. *Encyclopedia of Data Warehousing and Mining*. Idea Group.
- Gastwirth, J. L. (1972). The estimation of the lorenz curve and gini index. *The Review of Economics and Statistics*, 54(3):306–316.
- Ge, Y., Xiong, H., Tuzhilin, A., Xiao, K., Gruteser, M., and Pazzani, M. (2010). An energy-efficient mobile recommender system. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 899–908. ACM.
- Karlson, A. K., Brush, A., and Schechter, S. (2009). Can i borrow your phone?: understanding concerns when sharing mobile phones. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1647–1650. ACM.
- Melville, P. and Sindhvani, V. (2010). Recommender systems. *Encyclopedia of machine learning*, 1:829–838.
- Mooney, R. J. and Roy, L. (2000). Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204. ACM.
- Murphy, K. P. (2006). Naive bayes classifiers.
- Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *ICML*, volume 97, pages 412–420.