Privacy-preserving Content-based Recommender system for online social communities

Master Thesis von Pavan Kumar Merugu
Examiner: Prof. Dr.Max Müller
Supervisors: Aidmar Wainakh, M.Sc
Submission Date: 18. February 2019

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik
Telekooperation

Erklärung zur Abschlussarbeit gemäß § 23 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Pavan Kumar Merugu, die vorliegende Master-Thesis ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§38 Abs.2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei der abgegebenen Thesis stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung überein.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, den 18. February 2019

Pavan Kumar Merugu

Abstract

In recent years, the rapid growth of online social communities has led to a massive generation of user content. This user-generated content is required by the recommender systems to identify the interests of users without disclosing information to other parties participating in the system. Content-Based filtering (CBF) is one approach to identify the interests of users and recommend content that users have liked. In the process of content recommendation by recommender systems, users personal information may be exposed to potential privacy threats. To protect users privacy, a user group-based privacy-preserving recommender system is proposed, which is based on Elliptic curve cryptography system and Shamir's secret sharing scheme. In this system model, recommender server models users interests by determining interest similarities among users followed by association rule mining approach to generate content recommendations in a privacy-preserving fashion. We evaluate the proposed system model on publicly available datasets to measure performance and recommendation accuracy along with privacy-preserving theorems, which prove proposed system can protect users privacy.

Contents

Table of Figures

List of Tables

1. Introduction and motivation

One of the important activities of web users is Social Networking. Websites such as Facebook, Orkut, Instagram captured the interests of millions of people. Current Online Social Networking (OSN) Websites are based on centralized architecture and hence, called Centralized Online Social Networks (COSNs). COSNs provide web services which run on logically centralized infrastructure and provide a central repository for users and application data. Therefore, service providers in COSNs have control over the user's personal information such as posts, comments, photos, likes (possibly sensitive information) [1] . Various operations are performed on users personal data and thus exposing users privacy. Moreover, even after the agreement of legal policies by service providers, users personal information is exposed to third-party agencies to make recommendations [1]. Most recent consequences from a popular service provider: "*Facebook, reveals how users privacy is oppressed 87 million users profiles, which were collected over the years is handed over to a political firm "Cambridge Analytica*"."[1]
In turn, this stored data was used to build user profiles and interests to target advertising to gain political interests.
However, analyzing the privacy problems in current OSN seems to be fruitless and impractical, even if all the users are aware of the legitimate use of Social Networking Services (SNS), imposing appropriate privacy measures [2]. Authority of users information is still in the hands of service providers, which is a potential concern capable of exploiting users privacy. In the current situation, protection of users privacy is the primary objective, which current OSNs are not likely to provide.

The limitations in COSNs are addressed by designing an infrastructure, which decentralizes the control of authority from service providers in OSN. Unlike COSN, Decentralized online Social Network (DOSN) is designed on a peer to peer network architecture. In DOSNs, the concept of a single service provider is changed, where a set of peers shares the tasks required to run the system. Now, users do not need to register with a single commercial service provider instead they can choose the trusted peer to host their personal information or users themselves can host the services. With this approach, separation of authority and control from service providers is achieved and believed that users have more control over their data, particularly in these aspects [3]:
Privacy Protecting users privacy is considered as the key characteristic of DOSNs. Users are left with capabilities to choose the service host, where to store the information and whom to share the information. Additionally, information can be stored in chunks of data within multiple hosts to address a single point of failure problem.
Integrity Additionally, DOSNs should be able to protect users identity and information from tampering and modification. The users identity is not necessarily provided by a centralized server, but also by trusted parties in a distributed architecture.
Availability High availability ensures robust infrastructure against failures, exchange of messages. DOSNs should be able to provide continuous services to the users.

As discussed earlier, users of current OSNs are exposed to various privacy risks. Moreover, users information is used by SPs and data analytics companies to learn users behavior and interests for their own benefits. To overcome this drawback, researchers proposed DOSNs which are based on which are

[1] http://www.rmmagazine.com/2018/05/01/facebook-scandal-raises-data-privacy-concerns/

based on decentralized architectures, implemented on a network of trusted peers or peer to peer overlays [2]. In DOSNs, control of authority is moved from third-party SPs to users themselves. However, there are still challenges in adopting them. Social networks based on COSNs are widely used and the main challenge could be attracting a permanent user base. For instance, one of the popular DOSN-Diaspora [2], currently has about 669,000 users. Besides, being a popular DOSN, Diaspora is not well-established compared to OSNs such as Facebook, Twitter. Other limitations include, not everyone is interested to host the services on their computer. Moreover, managing DOSNs can be difficult for inexperienced users.

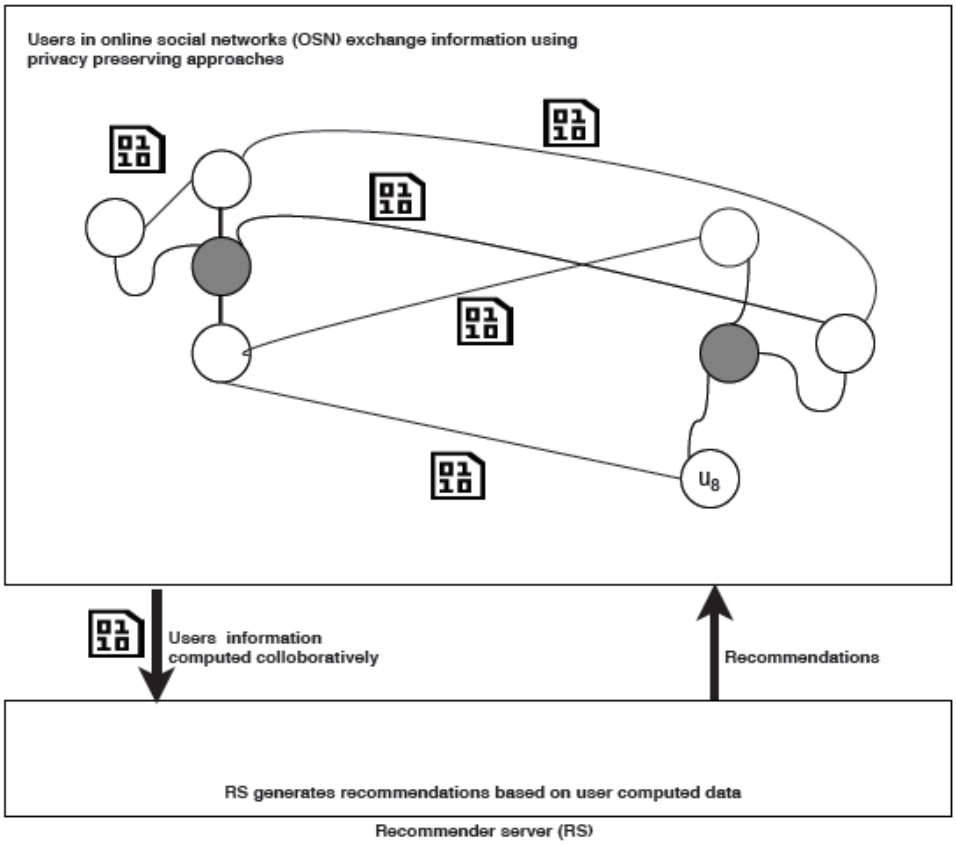


Figure 1. The general architecture of Hybrid Online Social Networks (HOSNs)

Above-mentioned limitations paved way for researchers to propose Hybrid Online Social Networks (HOSNs). Like DOSNs, users in HOSNs have control of their own information, what to share and whom to share, while enabling users to continue with existing SPs. A general overview of HOSNs is shown in figure 1. Users need not register with new OSNs to access services. As users are now able to continue with existing SPs which are based on COSN, their business model exists. Recommendations are one of the important activities of COSN and we should consider their business model prevails. Service providers can play role in recommendations without exploiting users privacy. Therefore, in HOSNs:

Users have complete control of their data, allows users to perform computations on their own data for recommendations, in a privacy-preserving fashion.
Service providers have no control over the management of users data. Therefore, SPs use data provided by users to perform recommendations.

Users in HOSNs collaborate to share the information to SPs. Users follow various approaches to hide the information from other users taking part in the online social community. SPs can only view nothing but the global information of all the users taking part in the OSN. However, computing the global information takes a lot of effort considering the distributive nature of the OSNs. There already exist various approaches which can perform privacy-preserving recommendations. But, current approaches lack in several factors such as efficiency, privacy and so on. To provide privacy-preserving recommendations, such factors as above should be considered while designing a HOSN system model.

1.1 Contribution

In the earlier section, we explained how users privacy is violated when recommending information (interests) to the users participating in online social communities. The goal of the thesis is to design and implement a recommender system for online social communities which protects users privacy. The contribution of the thesis is as follows:

- We design a system model which is able to generate high-quality recommendations to users in online social communities at the same time protect users privacy. To the best of our knowledge, this is the first work which performs association rule mining to generate recommendations for the users in online social communities in a privacy-preserving fashion.
- We use two cryptography protocols to effectively perform computations inside user groups. These protocols achieve privacy-preserving computations without revealing the identity of the users. Interest groups and pseudo-users are formed after performing computations using cryptography protocols.
- The system model is tested on two datasets popular datasets to evaluate recommendation quality, perform cluster analysis to choose an optimal number of clusters. The overall evaluation shows high-quality recommendations are achieved.

1.2 Structure of the thesis

The thesis is arranged in 8 chapters.

- Chapter 1 gives a brief introduction to the topic, a general overview of online social networks and privacy issues, how users information is stored in centralized servers to make personal recommendations to users, how the limitations of COSNs are overcome by DOSNs and a general overview of the solution to protect users privacy using HOSNs. At the end of this chapter, we describe the goals of this thesis.
- Chapter 2 gives an explanation of various privacy-preserving approaches which are used in building recommender systems. First, types of recommender systems are explained along with approaches used to filter data and make recommendations, then we discuss association rule mining, cryptography protocols, and clustering techniques, which are later used in this thesis to build our system model.
- Chapter 3 gives an insight into related work to the thesis: various privacy-preserving approaches, how researchers used this privacy-preserving approaches to build recommender systems or privacy-preserving models, what are the limitations in the existing works by various researchers.
- Chapter 4 includes the research problem related to the thesis and how we formulate the problem. This chapter gives an overview of the existing problems related to online social communities and a general overview of what should be done in order to protect users privacy.
- Chapter 5 gives the details of designing a recommender system for users in online social communities in a privacy-preserving fashion. The chapter is divided into four sections. The first section starts with describing building blocks of system design. The next sections give in detail description of user groups formation, user interest modeling, pseudo-users formation and delegation, and content recommendations.
- Chapter 6 describes how users privacy is protected in a system model. Privacy-preserving in user group formation, user interest modeling, pseudo user formation, and delegation, and final content recommendations is explained using theorems.
- Chapter 7 illustrates the evaluation results of the system against the two popular datasets. It explains the result with respect to *precision* and *recall*.
- Chapter 8 concludes the thesis and provides an outline for future work.

2. Background

The earlier chapter gave a brief overview of how users privacy is violated in online social communities. There exist solutions for privacy-preserving recommender systems. In this chapter, we go through the theoretical background of the already existing approaches. We discuss types of existing recommender systems and dive into privacy-preserving approaches employed in designing this recommender system. Some approaches include cryptography protocols, clustering based on similarity computations and frequent itemset mining to generate association rules. Later these approaches are used in designing our privacy-preserving system model for online social communities.

2.1 Recommender system

A recommender system provides a meaningful and useful set of recommendations to users based on information about user preferences [4]. The information gained from users can be derived explicitly or implicitly. Information used in recommender systems can be briefly categorized as [5]:

1) Behavioral Information is gathered while the user interacts with the recommender system. Therefore, information obtained is implicit. For example, product views on an online shopping website.
2) Recommendation feedback is response provided by the user to a recommendation or any item bought (or liked). Positive, negative, or something more descriptive is few feedbacks provided by users. Therefore, is explicit information. For example, the review given to a product bought in Amazon.
3) User preferences can be explicitly rated items on a scale of 1-5 stars or keeping a favorites list. For example, Netflix rating of a web series.

The recommendations can be any of any type: books, movies, restaurants, news and so on. Recommender system creates users profile from the information obtained as discussed above, to perform computations such as the degree of similarity, association rule mining. Recommendations are also often based on similar users or the relation between users and items. The prediction can generate recommendations to users in future based on what they liked in past [4]. Recommender systems can be classified into various types such as collaborative filtering, content-based, hybrid, knowledge-based and demographic-based. The most commonly used recommender systems are collaborative filtering and content-based [5]. Here, I briefly give an overview of proposed recommender systems above:

Collaborative Filtering (CF) [4] [5] is a widely used recommender system. In CF, content items are rated by each user. These ratings figure out the similarity between users like similar users like similar items or users like items that are highly rated. Similarity computation is performed using several metrics. In CF, an active user receives recommendations, which are highly rated by his most similar users or items that are rated as favorites. For example, Tapestry by Goldberg [5] is one of the first collaborative filtering recommender systems that was designed to retrieve email messages relevant to a user's interests from a mailing list called Usenet. CF generates recommendations based on past ratings of users. In CF, Users profiles are made available to the recommender server to run the recommendation process.

Content-based (CBF) [5]based recommender systems determine similarities between items to generate recommendations. They predict past users ratings and item features to generate recommendations while

CF uses previous ratings only. Item meta-data is used to compute similarities in CBF, unlike CF recommender systems. Examples of meta-data are a genre for music, action movies, political news.

Hybrid [5] recommender systems combine multiple recommender systems such as CF, CBF, Knowledge-based and so on. However, the combination of different recommender systems is not straight forward [6].

2.2 Distributed Privacy-preserving data mining

The key goal of distributed data mining is to perform computation on aggregated data values of all the participants in the system without compromising individual participants privacy. Computations are performed by participants collaboratively to obtain aggregate results and may not trust each other. Aggregated data may be distributed horizontally partitioned, vertically partitioned or a combination of both to achieve privacy-preserving distributed mining.

Horizontally partitioned [7][8] In horizontally partitioned data, the set of all attributes will be the same, but the number of transactions will be different at each site. In a fully distributed setting of horizontally partitioned data, each participant has private access to only their own data or attribute values. Applications of data mining such as clustering and association rule mining can be performed on this type of partitioned data.

Vertically partitioned [7][8] In vertically partitioned data, the set of attributes will be different for all sites, but the number of transactions will be the same at each site. A vertically partitioned approach can be extended to a variety of data mining applications such as k means clustering, decision trees, SVM Classification.

Hybrid partition [8][7]. In a hybrid distribution of data, data is distributed either first horizontally and then vertically or vice-versa.

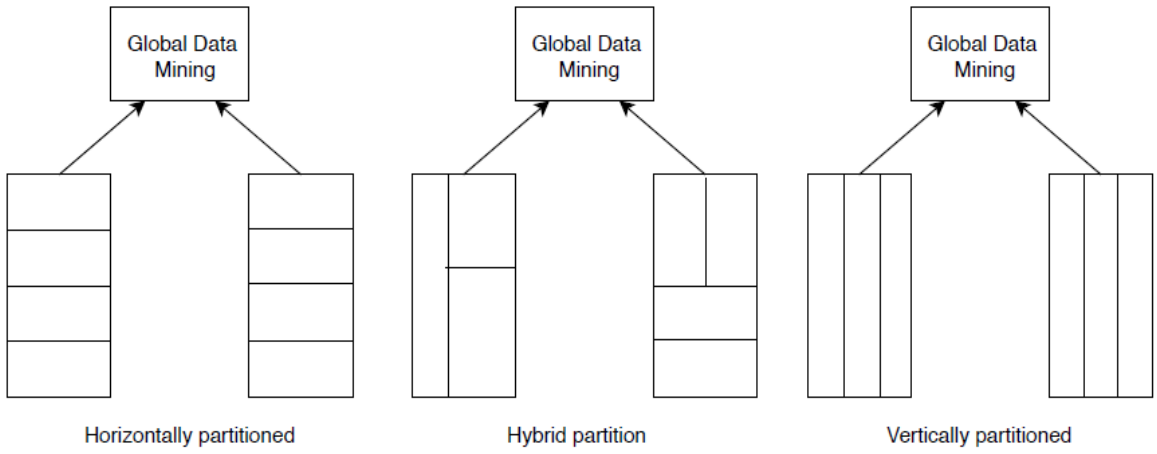


Figure 2. Classification of Data distribution

Several data mining techniques and algorithms are available to discover meaningful patterns and rules. These techniques have been discussed briefly in table 1 [9].

Approach	Description
Classification	Examines the features of newly presented object and assign it to a predefined class. For example, classify credit card applicats as low, high or medium risk.
Association	The main goal of association is to establish the relationship between items which exist in the market. The typical example of association modeling is Market basket Analysis.
Prediction	Unknown or missing attributes values are predicted based on other Information. For example, Forecast the sales value for next week based on avaliable data.
Clustering	In this form of datamining, data is organized into meaningful clusters such that attributes within the group are similar to each other, and as different as possible from the points in the other groups. It is an unsupervised classification.
Outlier Analysis	In this, Data Mining is done to identify and explain exceptions or deviations. For example, in case of MarketBasket Data Analysis, outlier can be some transaction which happens unusually

Table 1. A general overview of techniques in data mining

Secure multiparty computation

In Secure Multiparty Computation (SMC), parties collaborate to compute result but know nothing about each other[10]. Consider a trusted party to which all the involved parties send their input to compute output and receive final output. In SMC, parties perform computation without a trusted third party. All the parties collaborate to compute the result in SMC but with considerable communication overhead. It can be said that computation is secure if given a party input and output can be simulated to know what can be seen by the party. However, one must be careful while defining SMC because SMC computation will not reveal any sensitive data, but the resulting output may allow all the parties to deduce sensitive data from resulting data. Here the privacy of the system is violated if the parties can estimate the sensitive data.

Secure multiparty computation is the central problem with the cryptography protocols [11]. A problem of n parties is explained who holds private values (i_1, $i_2 \ldots i_n$) and wish not to expose private values to any other participating party but still want to compute function. Even in the presence of an adversary, controlling subset of parties to compromise privacy, the protocol should be able to find the correct result. The information released should be the result after computing values of all the parties. SMC protocols are difficult to design in case of the dishonest majority. However, recently many SMC protocols are proposed to compute results in presence of dishonest majorities. A well-established protocol that fits all the SMC computations is difficult to design.

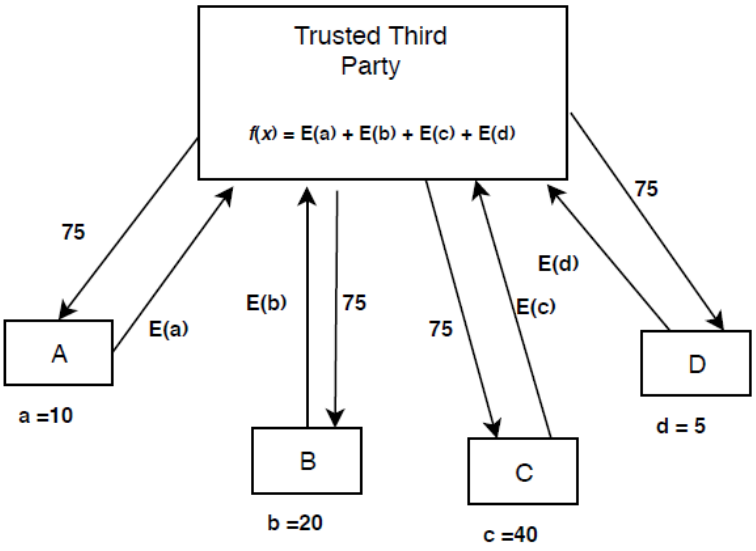


Figure 3. Secure multi-party computation using homomorphic property

A general multi-party computation (MPC) protocol can be designed, where a trusted third party is assumed to compute functions of multiple parties who wish to know nothing but the result after computation. For example, let *A*, *B*, *C* and *D* are the parties, want to compute a function. However, each party wishing to compute the result knows nothing about other parties' local values. First, each party sends their encrypted (signed) local value to the trusted third party. Then, the third-party compute function on the received encrypted local values (using homomorphic property). Therefore, the local values of each party are hidden from the third-party. At the end, each party receives the result and knows nothing about other parties involved.

2.3 Association rule mining

Association rule mining is used to discover rules that will predict the interesting relationships in large databases based on the occurrences of items in the transactions. Assume $I = \{i_1, i_2, i_3, ..., i_n\}$ is a set of size n binary-value attributes. Let Database DB = $\{t_1, t_2, t_3,...,t_m\}$ are transaction sets of size *m*. In this, each transaction t is called an itemset if $t \subseteq I$ [7]. A transaction *t* contains *X* if and only if $X \subseteq t$ and $X \subseteq I$. Then it is given as $X=>Y$ where $X \subseteq I$, $Y \subseteq I$ and $X \cap Y = \emptyset$ [7]. The support and confidence are given as.

$$Support\ (X=>Y) = \frac{|X \cup Y|}{|DB|}$$

$$Confidence\ (X=>Y) = \frac{|X \cup Y|}{|X|}$$

It is called as a frequent itemset if support value of an itemset is greater than or equal to user-defined minimum support threshold *s*[12]. Agarwal proposed frequent pattern mining for market basket analysis

and association rule mining[12]. The primary frequent pattern algorithms can be classified into two ways[12].

1) Generation of candidate sets.
   For example, Apriori algorithm
2) Without the candidate generation approach.
   For example, FP- growth algorithm

In this work, we focus only on Apriori algorithm for frequent itemset mining and generation of association rules.

Apriori Algorithm:

This algorithm uses prior information of frequent itemset and therefore the name Apriori. This algorithm works on iterative approach or level wise approach[12]:

1) In the first step, discover all itemsets from a given database that satisfy a user-defined minimum support threshold s. An itemset is frequent when its occurrence exceeds the user-defined minimum support threshold.
2) Assuming, all frequent k-itemset have been discovered, then create (k+1)-itemset based on k-itemset and keep just frequent (k+1)-itemset, i.e. a priori pruning operation is taken for excluding all infrequent (k+1)-itemsets.

The cost of mining rules in the first step is dominant because in this step the database needs to be scanned for counting the support value of itemsets[13]. Algorithm 1[14], describes the steps 1) and 2) in detail.

Algorithm 1: *Apriori*

Require: C_k: Candidate itemset of size k, F_k: frequent itemset of size k, *min_supp*: minimum support threshold

1: F_1 = {*frequent items*};
2: **for** k = 1
   While F_k not empty **do**
3: C_{k+1} = candidates generated from F_k;
4: increment k by 1;
5: **for** each transaction t in DB **do**
6: Increment the count of all candidates in C_{k+1} that are contained in t
7: F_{k+1} = candidates in C_{k+1} with *min_supp*
8: **end**
9: **end while**
10: **return** $\cup_k L_k$

For example [12], Let the minimum support threshold be 2. Given a set of transactions in table 1, our goal is to scan all the transactions to determine the count of each generated itemset and include only itemsets that have a count no less than minimum support threshold.

Transaction_id	Itemsets
1	A,B,C
2	A,C
3	A,B,C,E
4	B,C,E
5	D,E

Table 2. Transactions of itemsets in a database

Step 1: Finding candidate itemset C1 and large-itemset L1. Itemset D is eliminated as it does not achieve the minimum support threshold.

Itemset	Support
A	3
B	3
C	4
D	1
E	3

(C1)

Itemset	Support
A	3
B	3
C	4
E	3

(L1)

Table 3. Candidate generation 1 and Large-itemsets 1

Step 2: Finding candidate itemset C2 and large-itemset L2. Itemset A, E is eliminated as it does not achieve the minimum support threshold.

Itemset	Support
A,B	2
A,C	3
A,E	1
B,C	3
B,E	2
C,E	2

(C2)

Itemset	Support
A,B	2
A,C	3
B,C	3
B,E	2
C,E	2

(L2)

Table 4. Candidate generation 2 and Large-itemsets 2

Step 3: Finding candidate itemset C3 and large-itemset L3. Itemsets A, B, E and A, C, E are eliminated as it does not achieve the minimum support threshold.

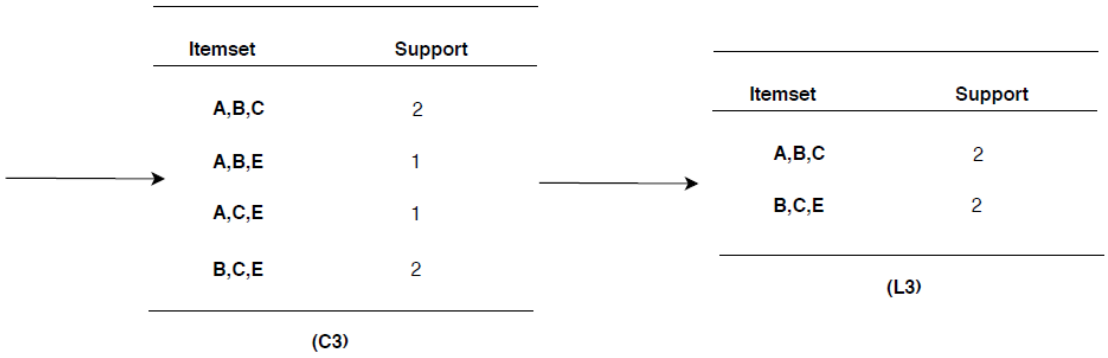


Table 5. Candidate generation 3 and Large-itemsets 3

2.4 Elliptic Curve Cryptography (ECC)

ECC is a public key cryptography system[15]. In this type of cryptography system each user or a device participating have a public key and a private key[16]. Also, each user or a device performs cryptographic operations associated with the keys [16]. The private key is always kept as a secret while the public key is shared with all participants taking part in communication. Unlike private key cryptography, the public key is much slower in terms of computation efficiency[16][15]. ECC is known to be an efficient cryptographic scheme compared to earlier cryptographic key such as RSA, DSA and DH[17] [18] [16]. For example, RSA uses large numbers for its operation. Therefore, larger would be the numbers in case of more security[15]. Basics of ECC is explained below [15]:

An elliptic curve 'E' is given by an equation. It is in the form of a curve as its name suggests:

$$E: y^2 = f(x) \qquad (1)$$

We make sure the curve is a non-singular and has no double roots. Therefore, the cubic form of the equation is:

$$E: y^2 = x^3 + a\,x + b \qquad (2)$$

To make the equation 2 a set, an extra point $\emptyset$ is added: “at infinity”.

$$E: y^2 = \{x^3 + a\,x + b\} \cup \{\emptyset\} \qquad (3)$$

Suppose, we want to find a point $P_2(x_2, y_2)$ on an elliptic curve and given a point $P_1(x_1, y_1)$. This can be calculated using point doubling such that $P_2 = 2\ P_1$.

$$x_2 = a + \lambda + \lambda^2$$

$$y_2 = (x_1 + x_2)\ \lambda + x_2 + y_1,$$

$$\text{where } \lambda = x_1 + \frac{y1}{x1} \qquad (4)$$

Now, if we want to find a point $P_3(x_3, y_3)$ on an elliptic curve and given two-point $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ derived from previous equations. This can be calculated using point addition such that $P_3 = P_1 + P_2$.

$$x_3 = a + \lambda + \lambda^2 + x_1 + x_2$$

$$y_3 = (x_2 + x_3)\ \lambda + x_3 + x_2,$$

$$\text{where } \lambda = \frac{y1 + y2}{x1 + x2} \qquad (5)$$

ECC implements two types of elliptic curve fields of interest defined over a finite field. They are [15]:

1) Prime finite fields and
2) Binary finite fields

Advantages of ECC [15]:

1) ECC uses much less key sizes compared to cryptographic conventions mentioned earlier.
2) ECC was generally implemented for low powered devices and therefore, it requires less power for its functioning.
3) It is more complex as scalar multiplication is used over multiplication or exponentiation infinite field.
4) ECC can produce a wide selection of elliptic curves and finite fields.

2.5 Secret sharing scheme

In this approach, a secret (can be local values of a participant) is shared among a set of participants. To reconstruct the secret key, k participants are required. Assume the secret S (S_1, S_2, S_3,..., S_n) is divided among P participants (P_1, P_2, P_3,..., P_n) where each participant is having a share of secret, respectively[14] [13]. Reconstruction of the secret key is possible only if enough shares are available. Therefore, the information cannot be obtained if a sufficient number of shares are not constructed [14] [13].

A secret sharing scheme works in two phases, distribution and reconstruction [14].

- **Distribution** In this phase the secret S is shared to P participants by computing the secret function to obtain a set of secrets ($[s]_1 \ldots [s]_n$). For example, $[s]_n$ is shared to participant P_n , where $n \geq i$.
- **Reconstruction** In this phase the secret S is obtained from a set of participants by reconstruction function. The set of participants computing reconstruction function are in the qualified set. Other

participants are forbidden from participating in the reconstruction phase also called as forbidden set [14].

For example [13]:

Distribution phase

- Assume secret S = 1456, p = 1615. Here, a subset of three shares are enough to reconstruct the secret S. Therefore, k = 3
- A random K-1 coefficient is selected. Here, the co-coefficients are 168 and 96.
- The polynomial $f(x) = 96\,x^2 + 168\,x + 1456$. Now, from the polynomial function $f(x)$, shares are generated for each participant.
- Shares are generated for each participant from polynomial function. Shares, s_1 = (1, 1720); s_2 = (2, 2176); s_3 = (3, 2824); s_4 = (4, 3664), s_5 = (5, 4696).

Reconstruction phase

- We need shares from three sites to reconstruct the secret key as k = 3 is chosen.
- Let three shares be, s_1 = (1, 1720); s_2 = (2, 2176); s_3 = (3, 2824). Lagrange polynomial is used here [13].

$$\ell_0 = \frac{x-x_1}{x_0-x_1} \cdot \frac{x-x_2}{x_0-x_2} = \frac{x-2}{1-2} \cdot \frac{x-3}{1-3} = \frac{(x-2)\,(x-3)}{2}$$

$$\ell_1 = \frac{x-x_0}{x_1-x_0} \cdot \frac{x-x_2}{x_1-x_2} = \frac{x-1}{2-1} \cdot \frac{x-3}{2-3} = \frac{(x-1)\,(x-3)}{-1}$$

$$\ell_2 = \frac{x-x_0}{x_2-x_0} \cdot \frac{x-x_1}{x_2-x_1} = \frac{x-1}{3-1} \cdot \frac{x-2}{3-2} = \frac{(x-1)\,(x-2)}{2}$$

Therefore,

$$f(x) = \sum_{j=0}^{k} y_j \cdot \ell_j(x)$$

$$= y_0 \cdot \ell_0 + y_1 \cdot \ell_1 + y_2 \cdot \ell_2$$

$$= 96\,x^2 + 168\,x + 1456$$

S = 1456 is the secret reconstructed.

2.6 Clustering and similarity measures

Clustering is unsupervised learning, where items in the cluster are as similar as possible to each other. Whereas items in one cluster are as different as possible to other clusters [19]. Therefore, clustering aims at grouping similar items to form a cluster using various similarity functions. A similarity function compares the values of data items to perform clustering. Next, we go through various categories of clustering [19] [20]:

- **Partition-based clustering** [19] Partition based clustering is the most often used clustering approach. This approach performs partitioning of data into *k* clusters, where each value in *k* represents a cluster. As discussed above, items within a cluster are "similar", and vice-versa. K-means clustering is one of the popular partition-based clustering approaches.
- **Hierarchical-based clustering** [19] In this approach the datasets are clustered or grouped based on the sequence of partitions. Grouping of items can be based on dividing or merging the datasets until all the items are split or grouped to form clusters. Grouping in hierarchical based clustering can be agglomerative or divisive. For example, REpresentatives (CURE) [19].
- **Density-based clustering** [19] In this approach, clustering is performed on a density basis using a threshold value. This approach can cluster items into arbitrary-shaped regions. An example of Density-based clustering is DBSCAN [19].

To calculate the similarity between data items, distance metrics plays a major role. Distance metrics function computes the similarity or distance between items of a set [19] [21] [22]. This helps in clustering items into groups which are similarly based on the results after computing similarity function. For example, assume set of items $I = \{i_1, i_2, i_3, i_4, i_5, i_6, i_7, i_8\}$ are in the dataset. Now, we perform clustering on the *I* based on the distance/similarity function. Figure. Shows clustering process on itemset *I*.

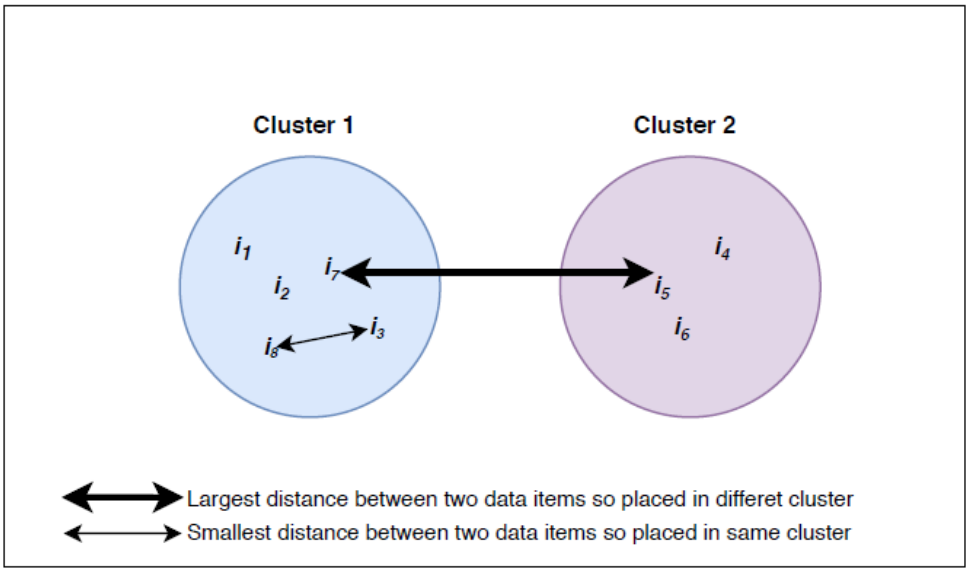


Figure 4. A generalized clustering performed on dataset [19]

It is also very important that items in the dataset are clustered efficiently. So, choosing an optimal number of clusters beforehand is an overhead [19]. Performing cluster analysis is the crucial part in efficiently clustering items of a dataset. A brief overview of general similarity function used for clustering is given below:

- **Cosine Similarity** [19] [23] is the measure of the similarity between two vectors based on the cosine of the angle between them. The angle zero represents similarity between vectors as one, the smaller the angle is, the more is the similarity. The equation of cosine similarity is given as,

$$Cos\_Sim(A,B) = cos(\theta) = \frac{A.B}{||A||\,||B||},$$

where A and B are vectors and the angle between A and B is θ [23].

- **Jaccard Similarity** [19] [23]is the measure of similarity between two sets of items, which is compared by a function intersection of both sets of items and divide by union of both. Jaccard similarity is given as,

$$JaccardSimilarity\,(A,B) = \frac{|A \cap B|}{|A\ U\ \boldsymbol{B}|},$$

3. Related work

In this chapter, existing approaches for privacy-preserving distributed association rule mining are discussed. Existing approaches can be classified into data perturbation approaches, which are further divided into addition and multiplication; secure multi-party computation, which is further divided into the secure union, secure comparison, and secure sum; and cryptography approaches, which are divided into Shamir's secret sharing, oblivious transfer and homomorphic encryption. Moreover, we describe briefly some of the existing approaches closely related to this work and discuss the limitations in comparison with this work.

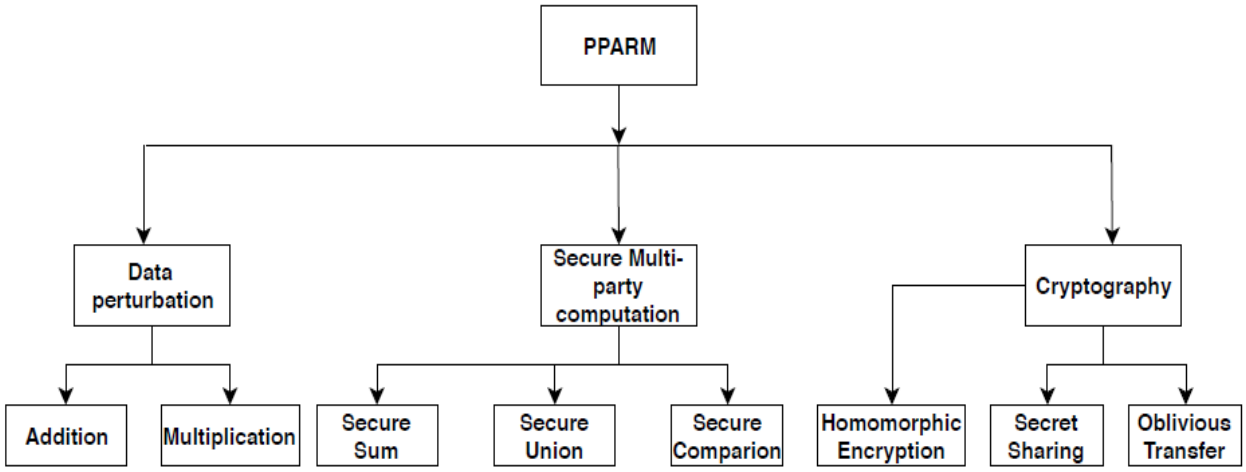


Figure 5. Privacy-preserving approaches in association rule mining [13]

Data perturbation techniques provide the privacy through modifying the original data values by adding and multiplying noise; later, it is exchanged with other sites. Hence, receiving sites are unable to identify the original data values. The basic idea of secure multi-party computation is that computation is secure. At the end of the computation, no site knows anything except its local value and global result. In secure sum method of secure multi-party computation, the initiator site chooses a random number uniformly and adds this to its local value and sends the sum value to next site; thus, the next site is unable to learn the actual local value of initiator site.

Chin-Chen Chang [24] proposed a privacy-preserving distributed data mining scheme, Enhanced Kantarcioglu and Clifton [25] Scheme's (EKCS) scheme. It is based on the [25]and works in two phases. In the first phase, EKCS reduces the quantities of encrypted global candidates and transmission load without any risk of privacy leakage. In the second phase, to prevent collusion two protocols are proposed in the communication environment.

Murat and Chris [26] proposed a method for Privacy-preserving distributed mining of association rules on horizontally partitioned data, that follows the basic approach where the values are passed between the local data mining sites rather than to a centralized combiner. The approaches are,

1) Frequent itemsets supporting in one or more sites
2) To verify if they have minimum support count threshold.

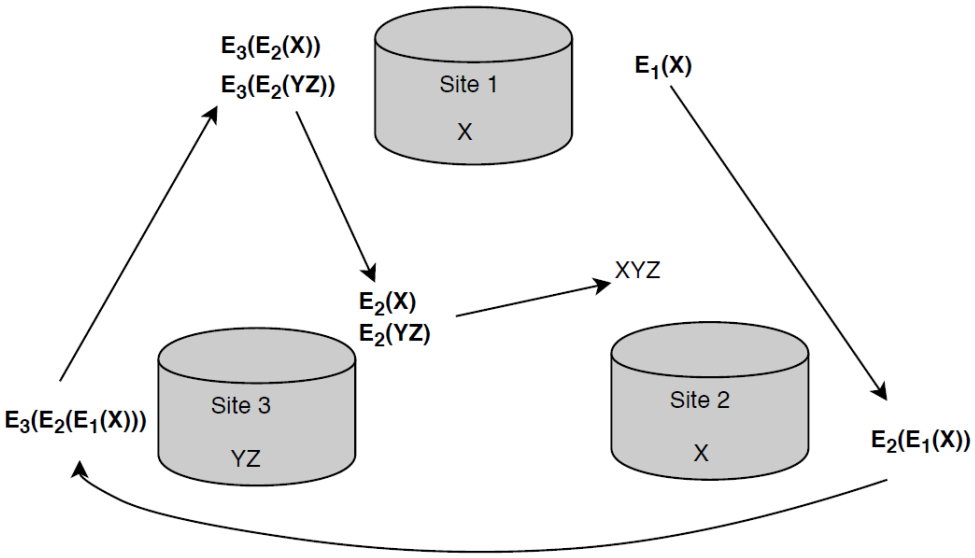


Figure 6. Determining global candidate itemsets [26]

The first phase uses commutative encryption for hiding source of itemset during the secure union of locally large itemsets. Each site encrypts its own itemsets that are frequent and then passes these encrypted itemsets to other sites. Then these encrypted itemsets are passed to decrypt and remove duplicates. Then, these encrypted itemsets are sent to a common site to eliminate duplicates if any, and to start the decryption process. Then, each site decrypts each itemset it receives. The result is the common itemsets (A and B are common result in the figure).

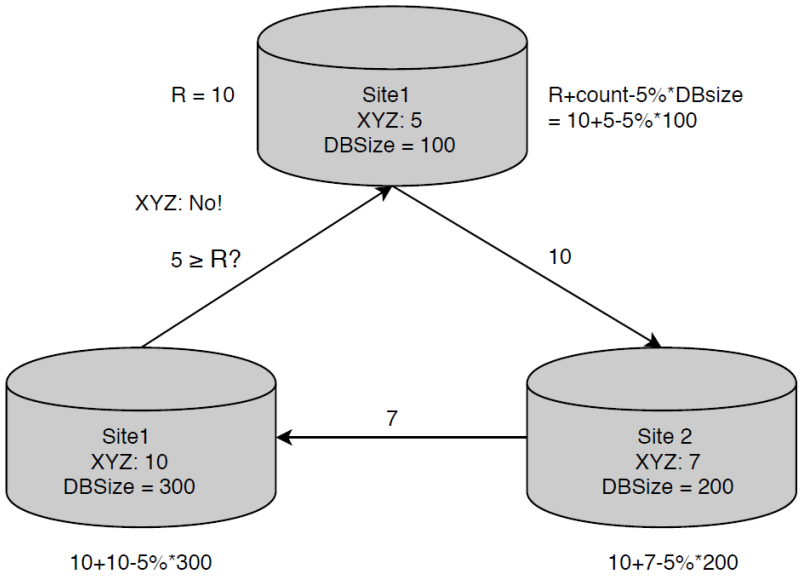


Figure 7. Determining if itemset support exceeds 5 percent threshold [26]

In the second phase (Figure 3), the secure sum is used to calculate global support count. In this phase, locally supported itemsets are verified to know whether they are supported globally. Local support counts are calculated on each site. In the figure, the itemset XYZ is known to hold support count at one or more sites. A random number *R* is chosen and added to the amount. This value is passed to the next site, which adds the count by which its support exceeds the threshold. This is passed to the last site in the figure, which adds its support again. The resulted value is verified to see if it has exceeded support threshold. If so, itemset XYZ is supported globally. It is a not a collusion-resistant protocol.

Dongsheng Li, Qin Lv [27] proposed "YANA" [27], a system model to preserve-privacy of users in online social communities. YANA automatically organizes users (with diverse content interests) into groups using a group construction protocol *SecureConstruct*. Users in the group collaborate to hide interests against recommender server. A user group in this approach has a set of pseudo-users. A unique interest is delegated to each pseudo-user generated and therefore, pseudo-users covers all interests in given a user group [27]. Recommender server interacts with real users through pseudo-users. Personalized recommendations are calculated on users side after receiving recommendations from the server [27]. Thus, users, private data is not exposed to the server. To ensure user privacy the authors in [27] proposed four SMPC protocols for in-group communication and computations.

The first protocol is a group construction protocol called as, *SecureConstruct*. As discussed above, this protocol automatically organizes users into groups in privacy-preserving and peer-to-peer fashion. Initially, a random user from the social community chooses to be the host of the group with the probability function mentioned in [27] "$Pr_{host}(u) = K_u/|U|$". where U is the set of users in the system and *Ku* is the expected user group size of the user group. If a user $u \in U$ is host then, he/she invites his/her friends to join the group with the probability function mention above.

After user grouping, the second protocol *SecureHash* is employed for user interest modeling. User interest is modeled by forming interest groups. Interest groups are formed after clustering similar items into clusters or groups. In this method, the k-centroids clustering method is adopted to cluster similar items. After interest groups are formed, each user goes through a set of items in the interest [27]. To estimate the distance between items a privacy-preserving distributed *MinHash* method is proposed. In the proposed hashing scheme, users perform multiple anonymous random walks to achieve random permutation of interests so that no one knows to whom the items belong. *HashVector*⟨ key, value ⟩ (a data structure) stores the hash values and the random walk stops after all the users have added their items to the data structure. Through anonymous communication protocol, the *HashVector* is sent to the server by the final user. After running the process for multiple random walks, the server will estimate the distance between items through 1 – Jaccard Similarity. After finding the distance between items, the server can cluster all the items into different user groups via k-centroid algorithm.

The third protocol *SecureSearch* finds the interests of users in a given group and helps in generation of pseudo-users. *The securesearch* algorithm is based on *SecureSum* protocol. In this approach, a user input value is divided into parts such that adding all the parts gives the final result[27]. Users obfuscate the values by sharing the values between themselves and send the sum of their local obfuscated parts to the "host". Host adds all the obfuscated parts and returns the sum to users. Therefore, no privacy of any participating user is revealed. Then, pseudo-users are generated based on the set of interests found after *SecureSum* protocol.

In the fourth protocol, *SecureRate* algorithm is proposed as no user would like to expose his/her interests, a privacy-preserving protocol is needed to maintain an interest profile for pseudo-users and item ratings

for each pseudo-user. Users in a given group run the *SecureSum* protocol as discussed in *SecureSearch*. This protocol creates pseudo-user profiles which are needed by the server to make recommendations. In the recommendation phase, the server collects the pseudo-users profile interests of all user groups. The server makes recommendations to pseudo-users by comparing similarities of pseudo-user profiles. After obtaining the recommendations. However, real users calculate personalized recommendations to obtain relevant recommendations.

As discussed above [27][24] [26] followed approaches such as secure sum, secure union, scalar product and secure size of the set to perform distributed association rule mining on horizontal and vertical partitioned data. This approaches hides the identity of the source of items but incurs huge computation cost. Moreover, sites can collude while computing the secure sum. In [27] during the recommendation process, the server can guess the interests of users based on similarity calculation among pseudo-users. Since the pseudo-users generated in the group is huge as each interest (item) has a pseudo-user assigned. There is an additional overhead of maintaining this pseudo-users

Shahriar Badsha, Xun Yi [28] proposed privacy-preserving protocol is to hide users' private information from the Recommender server RS, which generates recommendations and the Decryption server DS, which provides decryption services and privacy functions. They propose a new cryptographic protocol based (Boneh Goh Nissim (BGN)) cryptosystem by which secure multiplications can be computed by a single server. The private information in this system includes user ratings on items, user similarity, generated recommendations or any kind of intermediate computation results. No intermediate decryption is done to reveal messages to participants. Their approach is semi-honest, but participants are curious and usually do not collude with any other participant in the system. The proposed cryptographic protocol consists of two main phases:

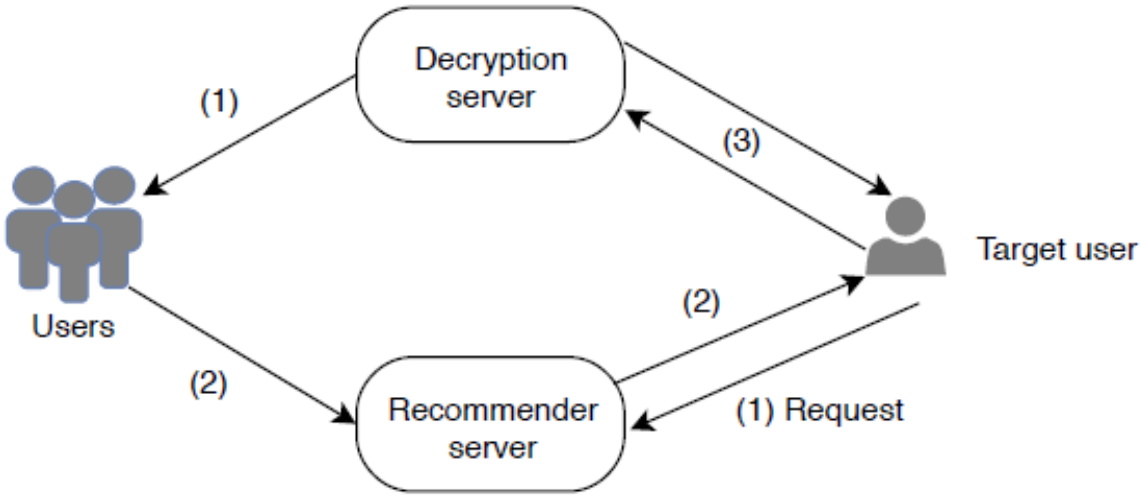


Figure 8. The general architecture of the proposed model [28]

Initialization phase

1) The DS generates public and private keys of the BGN encryption scheme and sends the public key to all participants.
2) All participants encrypt their ratings and send it to the RS for storage.

Recommendation phase

1) A user is also known as target user participates in the recommendation process by sending a request to RS. Encrypted ratings of the other users are received by the target user via the RS and locally determines the similarity in encrypted domain. The resultant ciphertexts are returned to the RS.
2) RS computes ciphertexts of recommendations based on the encrypted ratings of other users and encrypted similarities received from the target user. Once RS computes recommendations, it permutes the list of recommendations and signs the messages. Due to the permutation of recommendations, the DS is not able to identify the correct indices of items even after decryption. Moreover, by using the signature protocol, the DS can verify that the target user is not malicious nor sending any fake ratings and the ciphertexts of recommendations from the RS are authentic. However, the correct indices are required by target user so that he/she can reorder the list after getting the recommendations.
3) The target user sends the permuted list of ciphertexts with signatures to the DS for decryption. The DS decrypts the ciphertexts of recommendations by verifying the signatures. Corresponding item index with the highest recommendation result is sent to the target user. The target user locally reorders the item list and finds the correct item index as a recommendation.

To secure user privacy during recommendation process Badsha, Shahriar Yi, Xun Khalil, Ibrahim [4] proposed an efficient privacy-preserving item-based recommender system. The proposed system works in two phases:
In the first phase, all users compute average ratings of items by sending their rated items. Users encrypt their rating as well as flag information including zeros and send this ciphertext to the server to hide which items are rated. The server decrypts the users encrypted ratings and computes averages, similarities using homomorphic properties. All users in the system perform local computations and encrypt their item ratings. After server computing similarity, users can decrypt the ratings. The private information of users is not revealed during decryption.
In the second phase, recommender server computes recommendations using homomorphic properties based on similarities, average ratings, and target user's encrypted information and the target user decrypts this encrypted information using his own private key and gets highly recommended item from the decrypted results as recommendations.

The papers [4] [28] provide a good solution for protecting user privacy. In this approach, a target user computes recommendation process in a secure way using cryptography protocols. Their work provides accurate recommendations to users as each user calculates recommendations with the server. However, it is not an efficient solution for online social communities considering billions of users.

Chahar, Harendra Keshavamurthy, B. N. Modi, Chirag [13] have proposed two protocols for privacy-preserving distributed association rule. The first protocol, a digital signature based on Elliptic-curve-based Paillier public key cryptosystem is used, which is public key cryptosystem and needs shorter key

lengths compared to RSA, DH etc. Here a database DB is distributed among *n* sites $site_1$, $site_2$,..., $site_n$ and such that data in the DB and all the sites are horizontally distributed. Here, all involving sites are considered as semi-honest. As shown in figure 9, consider 4 sites $Site_1$, $Site_2$, $Site_3$ and $Site_4$ having the databases DB1, DB2, DB3, and DB4 respectively. Here $Site_3$ and $Site_4$ are combiner and miner. Certificate authority CA generates Elliptic-curve based Paillier public and secret keys and not responsible for storing any kind of information.

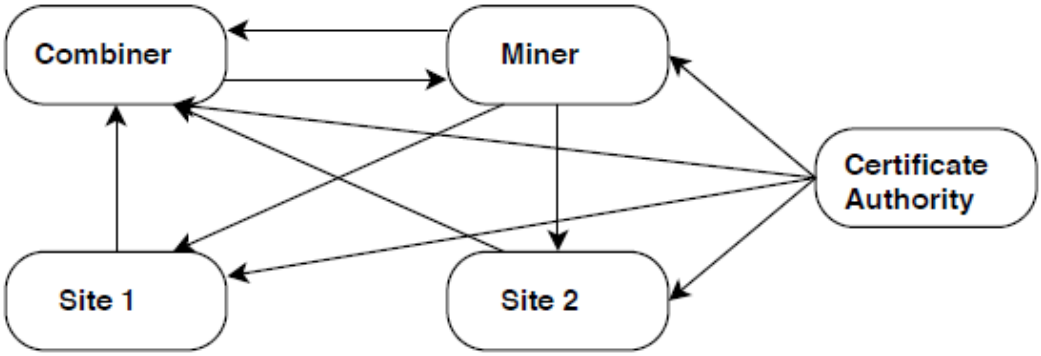


Figure 9. Proposed communication protocol [13]

- Each site generates local maximum frequent itemsets (MFI) and sends encrypted local MFI to the miner.
- Miner sends the local MFI to all the sites to generate all subsets from the set of local MFI at each site to find local support count.
- Each site sends encrypted local support of an itemset to combiner and combiner adds its local support to received encrypted local support of all the sites and sends it to miner using homomorphic encryption.
- Each site sends local database size in a similar way as local support of itemset.
- Miner then finds the global support count and frequent itemsets and broadcasts to other sites.

However, if miner and combiner collude the protocol fails. Therefore, the second protocol is proposed by Chahar, Harendra Keshavamurthy, B. N. Modi, Chirag to overcome this limitation in protocol 1. The second protocol "Shamir's secret sharing scheme" [13] addresses this limitation. CA generates public and private keys like in protocol 1 and distributes public key to all sites and secret key to respective sites except miner. CA generates different shares of the secret key of a miner and distributes them to respective sites. Miner needs shares from all the sites to reconstruct the secret key and decrypt the message. In this way, collusion between miner combiner is prevented. It is described that each site participates in association rule mining to generate frequent itemsets. In online social communities, users participating in computing association rules to generate recommendations would be slow as all the users are not expected to have computation power to perform association rule mining and therefore not an efficient approach for online social communities.

In the previous researches, we observed that high computations on the client side are required to generate recommendations. The previous works have assumed that the communication channel between the users and the server is safe. However, it not always true as an external adversary attack may affect the computation results performed by the recommender server and provide users with imprecise recommendations. Therefore, it is required to protect the communication channel between the entities to provide high quality recommendations. Some researches followed data perturbation techniques, where noise is added to the original values such that other users are unable to identify original values. However, this technique also degrades the recommendation quality if the original final values are not properly retained by the users.

4. Research problem

In this chapter, the research problem specific to online social communities is discussed and we formulate a general solution to achieve privacy protection of users information. The system model to be designed can protect user privacy by ensuring the communication channel between involving users is secure and adversaries will not be able to affect the privacy and security of messages exchanged between them. Adversaries monitor the communication channel between the users to affect the results. Therefore, an adversary attack will not affect the result on computation. If the results are affected by adversaries, the recommendations generated by the recommender system are not accurate as the result is computed on data which is affected. This system model developed is a group based as it organizes users into groups with diverse interests using a user group construction protocol same as in [27] so that each user's interests can be protected among a set of users who collaborate to distribute *itemLists* to the server. Here, *ItemList* is a set of items a user likes. After user group formation, a set of pseudo-users are formed. Pseudo-users are delegated to interest groups formed by server and combination of all interest groups covers all interests in a given group[27]. Recommender server is contacted by a set of pseudo-users to get recommendations. Real users obtain recommendations from pseudo-users, then calculate personalized recommendations based on pseudo-users recommendations. In this design, four privacy-preserving protocols are used for different in-group computations, which ensure user privacy.

4.1 Problem formulation

In this section, we first analyze the user interest privacy issues in online social communities and user-based recommender systems and then propose a high-level design for the proposed solution:

In online social communities (e.g. Facebook, Twitter, Google plus), users perform many activities[29]. Consider an online social community and its associated recommender server, the following operations are performed by any user in the online community[29]:

- Post or comment on an item shared or recommended by other users
- Read the comments and posts of other users and
- Finally, request recommendations from recommender server

From the above-mentioned operations, massive and diverse online content is generated by the users. Therefore, a challenge arises to protect users privacy from recommender systems. Further, public and private information of users in the online social community can be differentiated [29]. Users ‘‘*posts*’’ and “*comments*’’ are denoted public, as they are interacted with other users, while users “*read*’’ information is private as they do not intend to share with other users[27] [29].

Let a *u* has posted/read/commented on item *i*, we say u is interested (or liked item) *i*, then it is said that *u's* rating to *i* as $r_{i,u}$ = 1[27][29]. Otherwise, $r_{i,u}$ = 0[27][29]. Using binary ratings (“0” or “1”) [27], the recommender system can generate recommendations based on association rule mining approach and recommend items, which meet the minimum support count. In this work, only binary ratings of items from users are considered while other ratings such as 1-5 (example, Netflix movie ratings from 1-5) can still be supported [27]. For instance, 1–5 ratings can be changed to values between 0 to 1 by dividing the values by 5, so that “1, 2, 3, 4, 5″ will be calculated to “0.2, 0.4, 0.6, 0.8, 1.0″, respectively. We

change the ratings because to fit the binary ratings to our system model. For example, a user rating of an item greater than or equal to 0.6 indicates that the user is interested in the rated item.

In frequent itemset mining, recommender systems rely on users interests to mine items that appear often and recommend items that achieve minimum support value. The web browsing technologies such as virtual private networks trusted proxy, help users hide their IP addresses and provide no information to the online service providers [27]. However, these techniques do not help recommender systems to achieve accurate recommendations to the users [27]. So, a privacy-preserving recommender system must achieve accurate recommendations. To protect individual user's privacy, a high-level system design is proposed which is similar to [27]. The proposed design is supposed to provide recommendations to the users without sacrificing the content interest to any party participating in the system. Also, the modeled design targets large scale users in online social communities and is designed to be scalable and efficient. I will summarize key design goals of the model before discussing key components and construction of the model,

- Protect the privacy of all participating users who collaborate to hide their interests.
- Adversaries should not be able to affect the privacy and integrity of information passes through the communication channel.
- The design should be able to converge to a reasonable communication and computation cost.
- Users should receive high-quality recommendations.

5. Privacy-preserving Recommender system

In this chapter, we design a system model to recommend items to users in the online social community in a privacy-preserving fashion. First, we give an overview of the key components of the system design and then design the model based on key components described. The four-key sections in this chapter describe the construction of user groups for secure distribution of interests, modeling user interests to cluster similar content, the formation of pseudo-users to receive recommendations from the server, and finally content recommendation to generate recommendations from association rules. Note, the system model is similar to "YANA"[27]. We follow a similar approach as [27] in user group construction and user interest modeling. However, our system model deviates from [27] in pseudo user formation and content recommendation. During the recommendation process, "YANA" [27] follows the collaborative-filtering approach, whereas our system model generates recommendations based on association rule mining approach (content-based).

5.1 Design overview

As illustrated in Figure 1, the proposed design consists of four key components:

- **User groups** Users in the online social community are organized into user groups with a diverse content interest as discussed earlier. Users inside each group collaborate via privacy-preserving approaches such as elliptic curve cryptography and secret sharing, to protect users privacy from being violated by the recommender server. A host user of each group invites his/her friends to form a user group.
- **Interest groups** Inside each user group, interest groups are formed to find the true interests of users. Interest group identification ensures that users receive no "uninterested" items while receiving recommendations. In this system model, a k-centroid clustering algorithm is adopted to find the interest groups, which clusters similar items to form groups. Interest groups also help to select pseudo-users in a given user group.
- **Pseudo-users** On behalf of real users, pseudo-users interact with the recommender server to obtain recommendations. Each pseudo user in the user group is delegated to an interest group to obtain recommendations. The server makes recommendations to the pseudo-users based on their interests and users in the group re-calculate their personalized recommendations based on the importance of recommended items to them.
- **Recommendation algorithm** The server first needs to collect users *itemLists* to calculate recommendations. The secure distribution of users *itemLists* is achieved through efficient privacy-preserving cryptography approaches Elliptic-curve-based Paillier public key cryptosystem and secret sharing schemes. The combined *itemLists* of users in the social community allow the server to perform the proposed frequent itemset mining algorithm (Apriori algorithm) to generate association rules and make recommendations to the pseudo-users. Each real user, in turn, calculates his own recommendations from the pseudo-users.

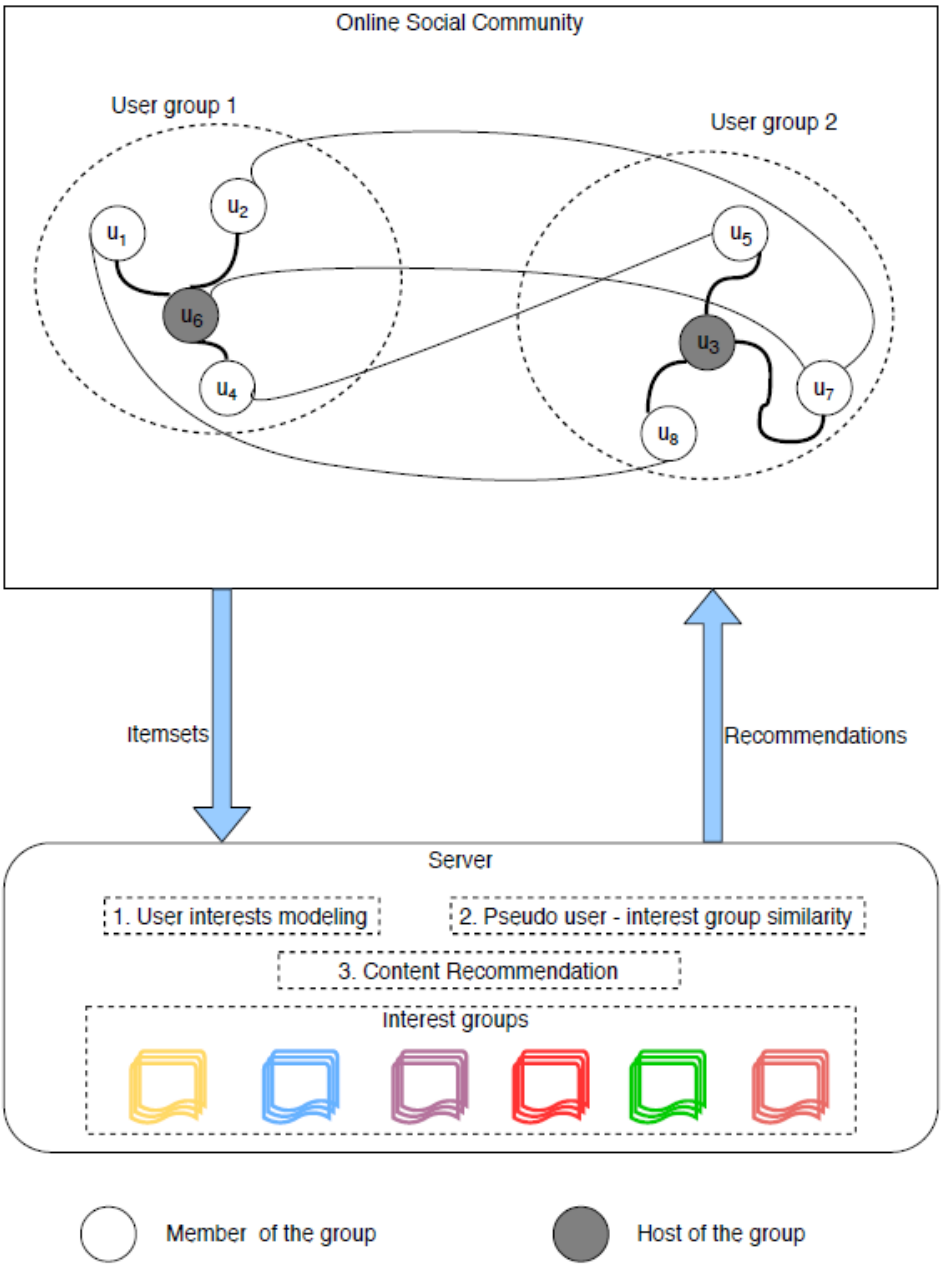


Figure 10. privacy-preserving content recommender system for online social communities

5.2 User group definition and construction

This section describes how user groups are organized in a privacy-preserving manner. Within a given user group, users collaborate and send *itemLists* to the server without sacrificing the privacy of any individual user.

Definition 1: A user group g is a three-tuple: $\{u_g, I_g, p_g\}$ where $g \in G$ in which u_g is a set of users who collaborate to form a user group and protect privacy of each other, I_g is the set of interest groups where each I_g in g contains items of similar content[27]. p_g is a set of pseudo-users, who get recommendations from the server. [27].

User groups can be formed in privacy-preserving fashion to hide the contents of each user from a set of users taking part in the same group. To organize users with various interests, a group construction protocol is proposed [27]. User group construction is shown in "*Algorithm 1*" [27] and for each user group g constructed, S_u is the number of users in a group and should be no less than 3. For instance, if the size of the user group u_g is 2 ($|u_g| = 2$)[27]. A user u_1 can easily infer the input values of user u_2. This a two-party model instance of jointly computing a result. This is explained in detail in chapter 2. Therefore, $|u_g| \geq 3$ (the size of the user group must be greater than 3). In the case of, $|u_g| \geq 3$ any computation performed by users in the given user group to infer the privacy of other users, will only be joint result of all the users in that group. Any privacy violated will only be the random guess of the users result. The user groups are constructed in a peer-to-peer way and therefore should be noted that users may choose to leave a user group or join another group for various reasons[27]. Therefore group requirements should be verified by other users in the group .i.e., if group size is greater than or equal to 3.

The *SecureGrouping* algorithm works as follows [27]:

- Let U be set of users who are participating to form a user group. Each user $u \in U$, the algorithm checks if the user is any user group already defined.
- If u has not joined any group, choose the host of a user group with a probability function $Pr_{host}(u) = S_u/|U|$ [27]. Here S_u is the size of the user group.
- If u is a host, the friends of u are invited to join the group.
- If a user u exists who has not joined any group, he chooses his friends who are the host of a user group and verifies that user group size is not empty. If the user group chosen is empty, he chooses another friend who is the host of a group and continues the same process until a user group is chosen.
- In case if user u has no friends who are hosts of a user group. Then he chooses a friend who already joined a user group and joins the group.

The complexity of user group formation

Construction of user groups is performed in a distributed way, where a host invites his/her friends to join the group. To join a group, the user chooses one of its friend who is the host of that group and joins his group. We assume that user group construction protocol terminates in certain random rounds [27]. Each user visits all his friends at least once. Therefore, the complexity is $O(|N_u|)$, where $|N_u|$ is the number of u friends [27].

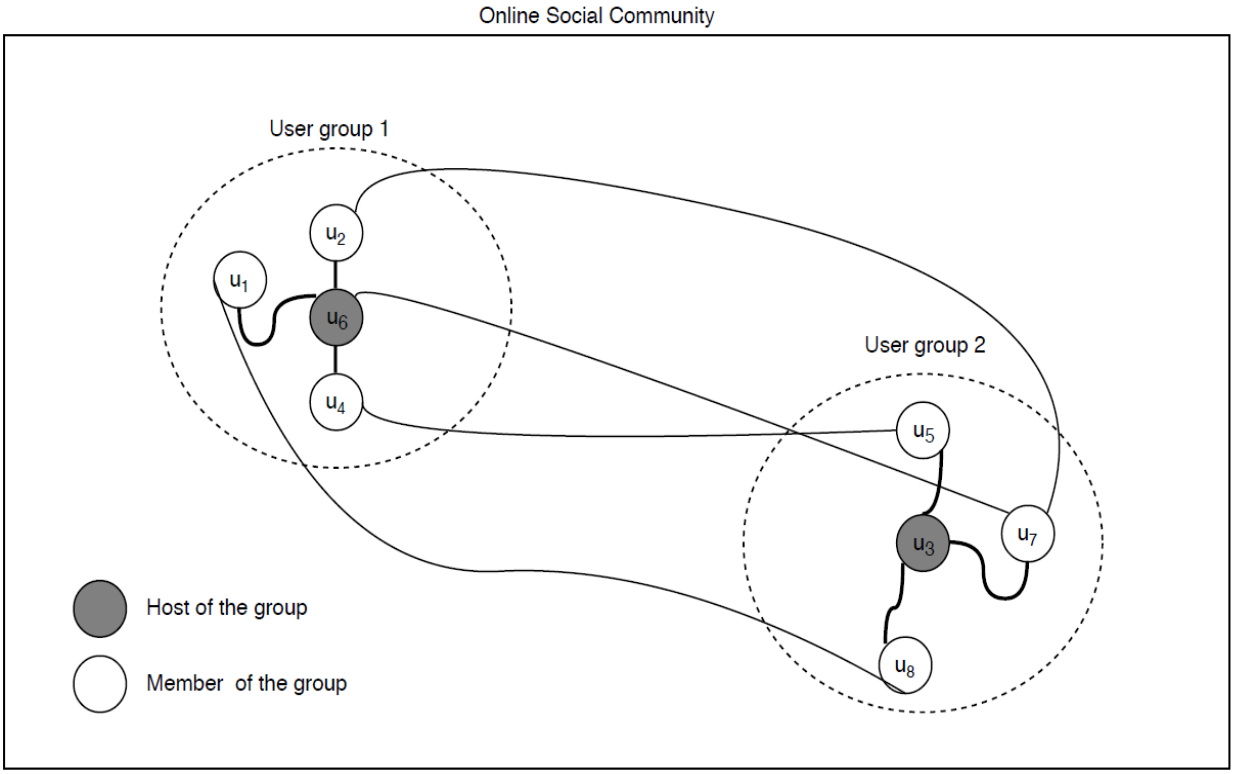


Figure 11. User group formation in online social communities

For example, let $U = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ be the users in the social community. A user from the social community chooses to be the host of a group with the probability $Pr_{host}(u) = S_u/|U|$. If a user u from U is host then, invites his/her friends to join the group [27]. In this example, let users u_3 and u_6 be the hosts of the group and invites their friends to join their groups. u_6 invites his set of friends $u_g = \{u_1, u_2, u_4\}$ to form a user group g_1 and u_3 invites his set of friends $u_g = \{u_5, u_7, u_8\}$ to form a user group g_2**.** Figure 2 illustrates, user group formation.

In the next sections, we discuss how interest groups are formed, how pseudo-users are formed and how these pseudo-users are delegated to interest groups.

5.3 Users Interest modeling: Interest groups definition and construction

This section describes how interest groups are formed in a privacy-preserving fashion.

Definition 2: $I_g = \{I_{g1}, I_{g2}, I_{g3}....I_{gk}\}$, where I_g is a set of interest groups and k is the number of interest groups, in which $I_{gk} = \{i_1, i_2, i_3, \ldots, i_m\}$ is a set of items and c_g belongs to I_g is the center of the group and represents "interest" of I_g and holds the property, for any two interest groups, I_{gi} and I_{gj}, where $i \geq 1$, $j \leq k$ and $i \neq j$, $I_{gi} \cap I_{gj} = \emptyset$ [29].

In this model, we cluster similar items into interest groups, similar to [27]. After interest group modeling, each user group will have interest groups distribution to generate pseudo-users. We choose k-centroid clustering method as in [27] [30]. The k-centroid algorithm chooses k items as cluster centers and groups items close to the close centers. The workflow of k-centroid clustering approach is shown in figure 12,

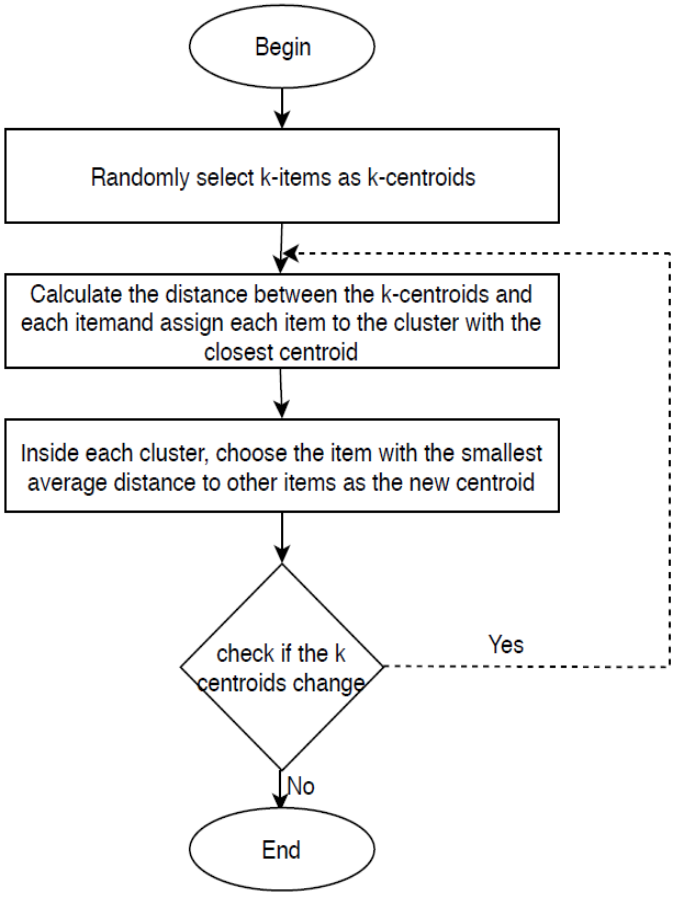


Figure 12. Workflow of the k-centroid algorithm

Challenges in identifying interest groups [27] [29]:

- An optimal number of interest groups, i.e. good inter-group separation and intra-group similarity. A better number of interest groups helps to generate accurate recommendations to users.
- Similarity computation of items in privacy-preserving fashion.

The privacy-preserving item distance calculation

Here, we discuss the main challenge to compute k-centroid algorithm in a privacy-preserving fashion. To preserve user privacy, two communication protocols are proposed. The first protocol is based on Elliptic curve cryptography [27]and the second one is based on "Shamir's secret sharing scheme" [27]. The proposed communication protocol using the two approaches is illustrated in figure 2. Once the server receives the itemLists of all the users in the social community, it can perform item distance calculation based on Jaccard similarity [27] [23]. It has the following property,

$$Jaccard\ similarity\ (i_1, i_2) = \frac{|i1 \cap i2|}{|i1 U\ i2|}$$

Which compares the similarity of the *itemsets* i_1 and i_2.

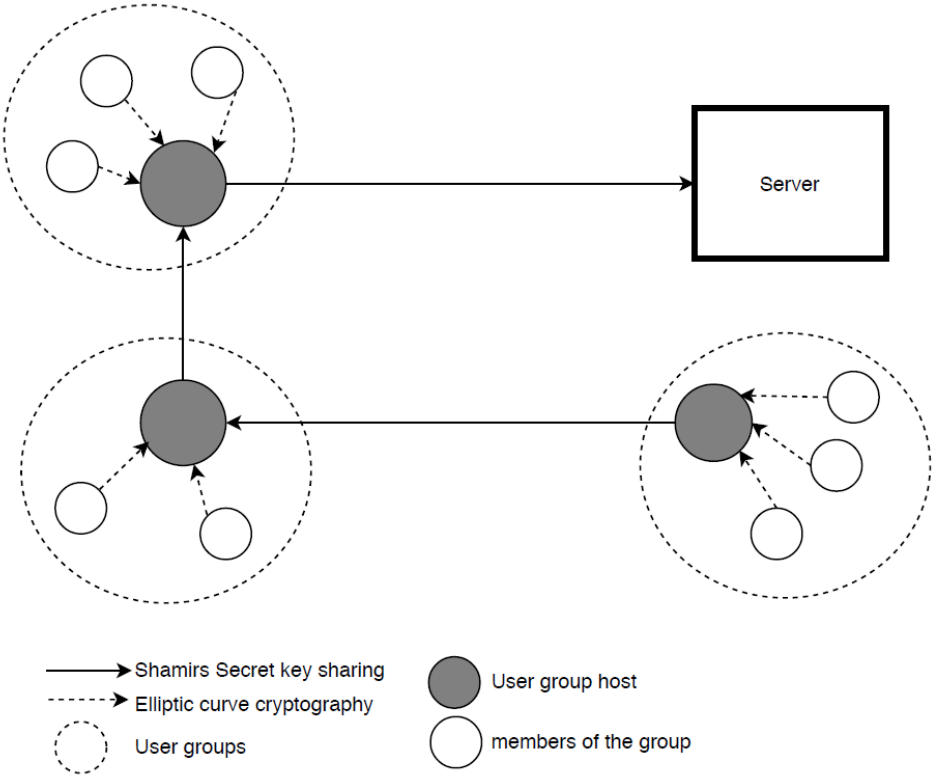


Figure 13. Proposed communication protocol

Phase 1 proposed protocol based on "Elliptic-curve cryptography"(ECC)[13][17]
Elliptic-curve-based Paillier public key cryptosystem is used in this phase as it requires shorter key length compared to RSA and Diffie-Hellman systems and saves significant computation time and memory space [16] [13] [31]. The messages are encrypted with the help of secret key of Elliptic-curve-based Paillier public key cryptosystem before sending it to hosts of the group. The authenticity and integrity of a message is kept via ECC[13].

- Certificate Authority (CA) generates public and secret keys to users, hosts, and server. CA distributes public keys of all users to other users and private key to respective users and a share of the secret key of the server to user group hosts and server.
- Each user in the group computes local interests, i.e. local *itemLists* and later each user encrypts the *itemLists* with the public key of the server and signs the encrypted *itemLists* with its own secret key. This encrypted and signed message is sent to the host of the group to which the user belongs.
- Each host receives all the signed *itemLists* from all the users in the group, and later verifies the integrity and authenticity of the signed message through the respective public key of users. It shuffles and combines the received *itemLists* with its own encrypted *itemLists*, signs the combined *itemLists* through its own secret key and later sends it to the predefined host.

Phase 2 proposed protocol based on "Shamir's secret key sharing" [13] [32]
To prevent the collusion between host and server, "Shamir's secret sharing scheme"[13] is used. As discussed earlier, the certificate authority (CA) distributes the public keys of each site to all other sites and distributes the secret key to respective hosts except the server. For reconstruction of the secret key, the server needs shares from all the hosts. Once, server reconstructs its secret key, it can decrypt *itemLists* of users signed (encrypted) by the public key of the server by all users, without revealing individual user *itemLists*. CA generates a polynomial, in which constant term will be the secret key of the server. Then CA generates different shares of the secret key of the server and distributes them to respective hosts. Now each host has one share of the secret key of the server. In phase 1, if the server and the host become malicious then they can collude with each other to reveal the *itemLists* of users. This is prevented using Shamir's secret sharing scheme since the server cannot decrypt the *itemLists* until it has shares from all hosts. For reconstructing the key, miner site needs shares from all sites, then it can decrypt the message. Thus, this approach prevents collusion of server and host.

The proposed protocol 2 works as follows: From the figure 2, u_6 and u_3 are hosts of user group 1 and user group 2, which have *itemLists* of the users u_1, u_2, u_4, u_5, u_7, u_8 along with *itemLists* of hosts of group u_6 and u_3, respectively. A polynomial degree k is generated by each host. The hosts also agree on distinct random values vector $X = (x_1, x_2, \ldots, x_n)$. Each host U_i chooses a random polynomial $p_i(x)$ of degree k, where $p_i(x) = I_i$ and $k = n-1$ [13]. Now, each host computes the shares of other hosts, including itself. Suppose host u_6 computes the shares, including itself as, share $(I_6, u_6) = p_6(x)$. Each host sends these shares to respective predefined hosts as share (I_6, u_6), here in this case to host u_3. Now host u_3 gets the share $p_6(x)$ and add the received share to compute $T(x) = p_3(x) + p_6(x)$. The result is sent to the server as the host u_3 is the last host. Thus, each host computes the global *itemLists* without revealing the local *itemLists* of real users. Once the server decrypts the global *itemLists*, it can estimate item distances based on Jaccard similarity between two sets.

The server can cluster all the items into interest groups based on the Jaccard similarity method with different cluster numbers – k. The optimal k is chosen by BIC score-based method [27]. The interest groups formed can help pseudo-users formation inside each user group and also helps In recommendation accuracy[27].

From previous example, let U be the user group in the social community and $U = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ be the users in the user group U. Let interests of each user in the user group be $u_1 = \{i_1, i_2\}$, $u_2 = \{i_3, i_4\}$, $u_3 = \{i_2, i_4\}$, $u_4 = \{i_1, i_5, i_6\}$, $u_5 = \{i_7\}$, $u_6 = \{i_1, i_2, i_5\}$, $u_7 = \{i_5, i_6\}$, $u_8 = \{i_2, i_8, i_9\}$. As discussed, server receives the *itemLists* of the users in social community securely through proposed communication protocol 1 and 2. Then, server estimates the distance between items using Jaccard similarity by building standard user-item matrix. After estimating distances between the items, k-centroid algorithm is performed to cluster similar items. Assuming that item-distance and clustering is performed on users items, derived interest groups could be $I_{g1} = \{i_1, i_2, i_4, i_7\}$ and $I_{g2} = \{i_3, i_5, i_6, i_8, i_9\}$.

Complexity analysis of user interest modeling

In user interest modeling, *itemLists* of users are distributed in privacy-preserving fashion using protocols Elliptic curve cryptography (protocol 1) and "Shamir's secret sharing scheme" (protocol 2). In protocol 1 n users encrypt their *itemLists* and send to the host of the group. Therefore, the complexity is $O(ne)$, where e is the cost of each user encrypting *itemList*. In protocol 2, n users who are hosts of the user groups collaborate to perform Shamir's secret sharing scheme. Therefore, the complexity of computing secret shares of received encrypted *itemLists* of users in the group is $O(n^2l)$. Here l is the cost of computing secret share among users (hosts). Once the server receives the encrypted *itemLists* from all the hosts in the social community it performs the decryption to perform clustering of items. Therefore, the complexity is $O(n^3d)$, where d is the cost of decrypting the ciphertexts of all the users. Then the server performs clustering of m items to form k clusters and the complexity is $O(k*m^2)$. The overall complexity of interest group modelling is $O(ne) + O(n^2l) + O(n^3d) + O(k*m^2)$. Even though the complexity is very high, most of the computation is performed on the server. Moreover, ECC is used which has shorter key lengths compared to other cryptography schemes and ECC can be used for low-powered devices which makes it an efficient protocol. Users only need to encrypt small bits of transaction sets (*itemLists*) in protocol 1 and in parallel. Considering the above reasons, the polynomial complexity is acceptable.

5.4 Pseudo-user management

After interest grouping, pseudo-users are formed in a given user group to protect real users privacy during the recommendation process. As discussed earlier, server interacts with the real users through the pseudo-users, i.e. recommendations from the server are published to the respective pseudo-user. Each Pseudo-users is delegated to an interest group or associated with an interest group in a given group. The real users calculate recommendations from pseudo-users.

Pseudo-users formation and delegation

Pseudo users are formed inside each user group, based on the interest groups modeled in the previous section. Each user group obtains set of interest groups calculated by the server. Then, users in the same user group construct a set of pseudo-users, each of which "delegates" a unique interest group. For each interest group delegated to the pseudo-user inside a group, the pseudo-user profile is maintained, which is nothing but the *itemLists* of interest group associated. The pseudo-user profile is required by the server to recommend itemsets to the respective pseudo-user to whom the recommendation belongs.

Given a set of interest groups, our goal is to associate pseudo-users to the corresponding interest group. *Algorithm 2* [27], illustrates the formation of pseudo-users in a given user group. It is similar to user group protocol mentioned in [27].

Algorithm 2. SecurePseudoUSer (g, I_g)

Require: Ug is the set of users in a given user group g, I_g is set of interest groups.

1: $P_g = \emptyset$;
2: **For** each user, $u \in U_g$, g_u is the expected user group size of u;
3: **while** Not any user in U_g are pseudo-users **do**
4: **for** each $u \in U_g$ who is not a pseudo user **do**

5: **for** each interest group $i_g \in I_g$ **do**
6: u chooses to be the "pseudo-user" of an interest group with probability $Pr_{pseudo}(u) = g_u/|U_g|$
7: **if** *u* is pseudo user **then**
8: chooses another user in the group to be a pseudo user;
9: **end if**
10: Assign user u as the pseudo-user for the interest group
11: $P_g = \{u\}$;
12: **end for**
13: **end for**
14**: return:** P_g;

Complexity analysis of pseudo-user formation and delegation

During pseudo-user formation, the users U_g in a given user group *g* will be visited once to check whether they are already pseudo-users or not. Therefore, the complexity of pseudo-user formation is $O(|U_g|)$ [27]. Now each pseudo-user is delegated to an interest group. Therefore, each pseudo-user visits all the interest groups at least once to check which interest groups are not delegated. Therefore, the complexity of pseudo-user delegation is $O(P_g * I_g)$ [27].

5.5 Content Recommendations

Content recommendations are performed on server-side and client-side. The server makes a recommendation to pseudo-users and real users, in turn, calculate recommendations from pseudo-users based on their preferences.

Server-side recommendations

The server requires a standard user-item rating matrix, which is required in association rule mining to generate association rules which are considered as recommendations. From section 5.3, it is known that the server receives *itemLists* from all the users securely through communication protocols 1 and 2. Therefore, the server can construct a user-item rating matrix without knowledge to which user the *itemList* belongs.

TID/ items	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9
U_1	1	0	1	0	0	0	0	0	0
U_2	0	1	0	0	0	1	0	0	0
U_3	0	1	1	0	0	0	0	0	0
U_4	1	0	0	0	0	0	1	1	0
U_5	0	0	0	0	1	0	0	0	0
U_6	1	0	1	0	0	0	1	0	0
U_7	0	0	0	1	0	0	1	1	0
U_8	0	0	1	1	0	0	0	0	1

Table 6. Candidate generation 3 and Large-itemsets 3

In the recommender server, Apriori algorithm is employed for discovering association rules to recommend items to users. The server has the user data, unable to identify to whom the transaction sets belong to. Thus, the server will calculate the association rules from the obtained matrix of users-items and makes recommendations to each pseudo user associated with an interest group[9]. As discussed in chapter 2, association rule mining is described as:

An association rule is an implication in the form of $A \Rightarrow B$, where $A, B \subset I$ are itemsets, $T \subseteq I$, where T is a transaction set containing list of items and D be a database with different transactions T_s and $A \cap B = \emptyset$. A is called antecedent while B is called consequent, the rule means A implies B. The basic measures of association rules are support (s) and confidence(c). Association rule mining is to find out association rules that achieve minimum support count and confidence from a given database.

From the previous example, let g be the user group in the social community and $U = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ be the users in the social community. Assuming, interests of each user be $u_1 = \{i_1, i_2\}$, $u_2 = \{i_3, i_4\}$, $u_3 = \{i_2, i_4\}$, $u_4 = \{i_1, i_5, i_6\}$, $u_5 = \{i_7\}$, $u_6 = \{i_1, i_2, i_5\}$, $u_7 = \{i_5, i_6\}$, $u_8 = \{i_2, i_8, i_9\}$. The server receives the itemLists of the users in the social community securely through employed communication protocol 1 and 2 from the previous section. Table 6, shows the user-item matrix where '1' denotes that user likes the item and '0' denotes user does not like the item. However, each row in the user-item matrix does not belong to the real user in the social community. Therefore, transactions of user likes are built. As discussed earlier the server has no knowledge of real users itemLists.

The server after generating the user-item matrix performs the Apriori algorithm to generate the association rules for the content recommendation. In the above example, let the minimum support threshold be 2 and length of association rules generated be 2.

Step1: Server then creates a candidate itemsets table for all the items along with the support count. In this case, table 7. Shows candidate itemsets C_1 that have been generated. The transaction values in table 6 are scanned to check support count of corresponding itemsets.

Items	Frequency
i_1	3
i_2	2
i_3	4
i_4	1
i_5	1
i_6	1
i_7	3
i_8	2
i_9	1

Table 7. Generation of Candidate itemsets C_1

Step 2: Items selected are only those which achieve minimum support threshold. Here, the support threshold is 2 as discussed above. Items that hold the minimum support count are i_1, i_2, i_3, i_7, i_8. Table 3. Shows Large-1 L_1 itemsets. Itemsets whose support counts are less than the pre-defined threshold is eliminated.

Items	Frequency
i_1	3
i_2	2
i_3	4
i_7	3
i_8	2

Table 8. Generation of Large-1 L_1 itemsets

Step 3: In the next step, all the pairs which are frequent and support minimum support threshold are selected. Here the order of the itemsets does not matter. Pairs of itemsets are selected by pairing first item with other items in the list, such as i_1i_2, i_1i_3, i_1i_7, i_1i_8. Now, consider the second item and pair it with preceding items, i.e., i_2i_3, i_2i_7, i_2i_8 and similarly, continue the same process for third and fourth itemsets i_3i_7, i_3i_8, i_7i_8. So, all the itemset pairs in this example are i_1i_2, i_1i_3, i_1i_7, i_1i_8, i_2i_3, i_2i_7, i_2i_8, i_3i_7, i_3i_8, i_7i_8.

Itemsets	Frequency
i_1i_3	2
i_1i_7	2
i_7i_8	2

Table 9. Generation of Large-2 L_2 itemsets

Step 4: From the itemsets achieved in the previous step, the algorithm verifies the support of each pair in all the transactions and only those itemLists which cross the minimum support threshold are considered.

Therefore, table 4 shows the L_2 itemsets generated.

Step 5: Till now, frequent itemset generation is performed by the server using the Apriori algorithm. In the next task, we see how to find the association rules efficiently. As discussed earlier the maximum length of association rule generated is given as 2. We find that i_1i_3, i_1i_7, i_7i_8 are the frequent itemsets that achieve the minimum support threshold and maximum length. Furthermore, the algorithm terminates

here because the generation of L_3 itemsets is not possible as no transaction achieves minimum support threshold. Therefore, the association rules are $i_1 \rightarrow i_3$, $i_1 \rightarrow i_7$, $i_7 \rightarrow i_8$, which indicates users who are interested in item i_1 are also interested in item i_3 and so on for other two association rules derived. Therefore, the server recommendations are shown in table 5,

Recommendations	
R1	$i_1 ==> i_3$
R2	$i_1 ==> i_7$
R3	$i_7 ==> i_8$

Table 10. Association rules for recommendation

Client-side recommendation

The item recommendation from the server is delivered to respective pseudo-users to whom the item belongs. Real users then calculate their personalized recommendations of items-based pseudo-users recommendations. As discussed, server-side recommendations contain antecedent and consequent. For instance, $i_1 \rightarrow i_3$ where i_1 is called antecedent while i_3 is called consequent, the rule means i_1 implies i_3. This association rule generated is passed as a recommendation to the real users through pseudo-users. A real user verifies if the antecedent is present in his itemList. If present, the user gets the consequent as a recommendation. If a real user does not have the antecedent in his *itemList*, the recommendation is ignored as it is not important to a given user. Rating of a recommendation (item) is calculated as,

$$Rating\ (r_i) = |I_u \cap I_{r,\ antecedent}| \quad (2),$$

Where I_u is a set of items a user likes $I_{r,\ antecedent}$ is a set of items recommended to a user. If $r_i > 0$, the user gets the recommendation. Otherwise, the recommendation is not received by a real user.

Complexity analysis of the recommendation process:

The recommender server performs frequent itemset mining to generate recommendations. From section 4.4, the server receives all the *itemLists* of users in the social community. Now, the server performs Apriori algorithm for generation of recommendations. Assume there are *m* unique items while computing Apriori, items are visited step-wise know as candidate generation and large itemset generation. For each step computed, a subset of items is visited. Therefore, the complexity of server-side content recommendation is $O(2^m)$. server recommends *m* recommendations to set of pseudo-users p_g in the system and the complexity is $O(m*p_g)$. On client-side, recommendations are calculated b real users and the complexity is $O(|p_g|)$, as they calculate from a set of pseudo-users.

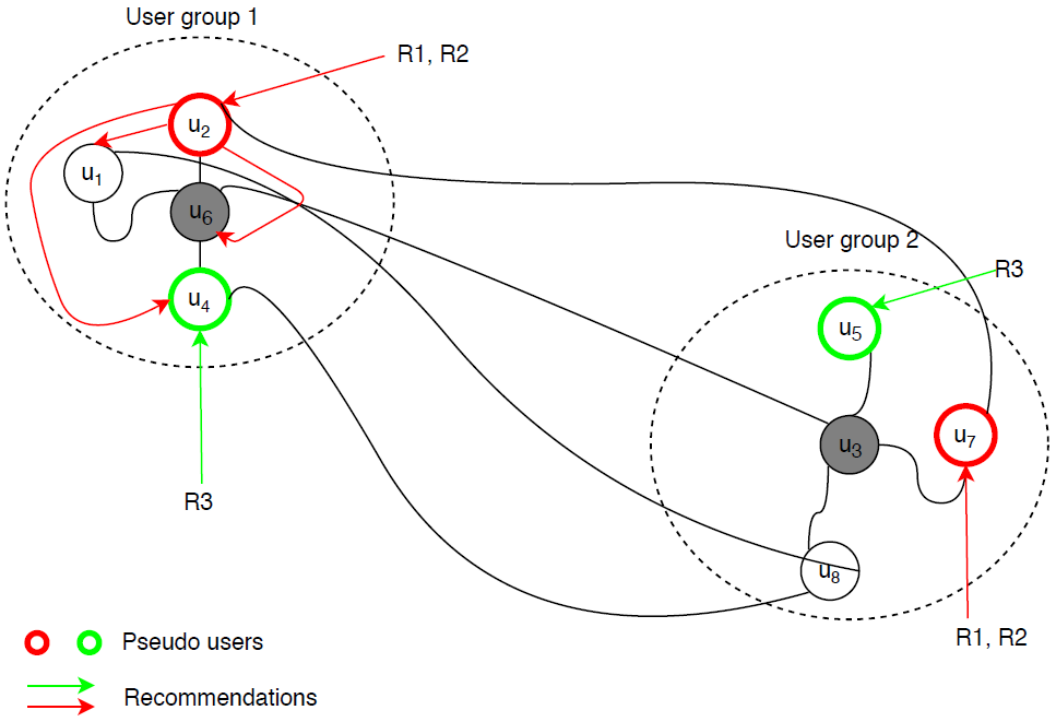


Figure 14. Example client-side Recommendation

For example, the figure shows the recommendation process in user groups 1 and 2. Recommendation *R1* and *R2* is received by respective pseudo-users u_2 and u_7 in the user groups 1 and 2. Each real user in the group calculates r_i of the items recommended with a pseudo-user from the equation 2. Here in this example, users in group 1receive *R1* and *R2* as $r_i > 0$, while no user receives *R3* in group 1 as $r_i = 0$. In group 2, *R1*, *R2*and *R3* are not received by any user as $r_i = 0$.

6. Privacy-preserving in the system model

In the previous sections, technical details of the model have been presented. It is seen that users privacy is preserved by a group of users collaborating to hide individual users privacy. In this section, we discuss privacy protection in each phase of the model.

6.1 Privacy protection in the user group

"Theorem 1" [27]. Consider an online social community with a set of users U ($|U| > 1$). The interest privacy of users in U will not be revealed during the execution of *SecureGrouping* (Algorithm 1).

Proof. During the execution of *SecureGrouping*, users do not share any information to join a user group. The only communication needed is whether a user joined a user group or not. Therefore, the *SecureGrouping* algorithm is privacy-preserving.

"Theorem 2" [27]. A user group g is constructed using *SecureGrouping I* algorithm and $S_g \geq 3$, where is the size of user group constructed. Suppose a user u_1 computes a similarity function to infer the interest privacy of user u_2 in the same user group. The users follow a semi-honest behavior. A user u computing similarity function on user u_1 and user u_2 are perfectly indistinguishable.

Proof. Let $i \in I$ (where I is set of items in the social community) be the set of items satisfying $r_i > 0$ in g, $r_i > 0$ means items liked by users in a given user group. Because all the other intermediate information is encrypted using communication protocols 1 and 2, no other information is obtained by users during computation and users are non-colluding, ensured by communication protocol 2 [27].

For each item i ($r_i > 0$), the user performs computation on input and output of i. The dis-honest users perform computation on input and output value of i in a given user group. Consider u, u_1 and u_2, where u is a dishonest user and computes on input and output values of u_1 and u_2. The given two scenarios explain the theorem:

- If the result computed on input and output value is '0', that means no user is interested in i, so that u cannot infer any information as the result computed is nothing but '0'

- If the result computed on input and output value is '1', that means users u, u_1 and u_2 likes the item i. Also, all the users might have liked the item i or only a few users might have liked the item. So, u cannot know whether u_1 liked i or u_2 liked i. Any information u inferred from the result of output is the random guess.

Therefore, in both the cases, the result is indistinguishable[27].

6.2 Privacy protection in user interest modeling

In this section, we prove that the user interest modeling method protects users interest from adversary attacks and is a privacy-preserving. The first protocol is based on Elliptic-curve cryptosystem, while the second protocol is based on "Shamir's secret sharing scheme" [13].

Theorem 3 [27]. Let there be set of users *U* taking part in the online social community and ($|U| > 1$). Then we prove that the modeling users interest is privacy-preserving.

Proof. Communication protocol 1 and protocol 2 are the two crucial steps required for item similarity computation. Users transactions are distributed to the recommender server in a privacy-preserving fashion. After distribution of users transactions, recommender server calculates item distance to cluster items into interest groups. Here, communication protocol 1 and protocol 2 are discussed in two phases and prove they are privacy-preserving:

Phase 1 In this stage, each user u in user group sends his item list to the host of the group. If u chooses not to add its *itemList* then the output of u is an empty list. Therefore, no *itemList* is sent to the host of the group. In this protocol, Elliptic-curve-based Paillier public key cryptosystem since it requires shorter key length and provides the same level of security as discussed earlier. Each user u signs the *itemList* with the help of Elliptic-curve-based Paillier public key cryptosystem before sending it to host which in turn helps in validating the integrity and authenticity of *itemList* sent by each user in the system. A host of the group receives all the signed *itemLists* from all the users in the group and verifies the integrity and authenticity of signed *itemLists* through respective public keys of users. Also, host shuffles and combines the received *itemLists* with its own signed *itemList*. Later combines the signed *itemLists* of all the users in the group with its own secret key and send it to the next host. Here, certificate authority does not have any database part and generates the Elliptic-curve based Paillier public and secret keys for all the involving users in the social community. we see that the homomorphic property of Elliptic-curve-based Paillier cryptosystem helps find the *itemLists* of all the users in the group securely. The proposed communication protocol works as follows:

For each *itemList I* that belongs to *(n - 1)* users, the users *itemLists* in the group can be derived as follows [13]:

Encryption: $E(I_1 + I_2 + I_3 + I_4 + \ldots + I_{n-1}) = E(I_1) * E(I_2) * E(I_3) * E(I_4) * \ldots * E(I_{n-1})$

Decryption: $D(E(I_1 + I_2 + I_3 + I_4 + \ldots + I_{n-1})) = I_1 + I_2 + I_3 + I_4 + \ldots + I_{n-1}$

After the decryption process, the result will be equal to combined *itemLists I* of all the *(n-1)* users in the user group.

The proposed communication protocol 1 securely collects the *itemLists* of the users in a group, since all the information is performed after performing encryption and signing. Also, ensures the integrity and authenticity of the received information. However, the *itemLists* are not sent to the server as it may fail if the host of a group colludes with the server. There Shamir's secret sharing scheme is employed to prevent collusion.

Phase 2 The proposed Shamir's secret sharing scheme helps to prevent collusion between host and server. The certificate authority distributes the public keys of each host to all other hosts and distributes

the secret key to respective host except the server. It generates a polynomial, in which constant term will be the secret key of miner site. Then it generates different shares of the secret key of the server and distributes them to respective hosts. Now each host has one share of the secret key of miner site. In protocol 1, if server and host become malicious then they can collude with each other to reveal the *itemLists* of other users in the system. This is prevented using Shamir's secret sharing scheme since the server cannot decrypt the *itemLists* until it has shares from all hosts. For reconstructing the key, the server needs shares from all hosts, then it can decrypt the *itemLists* of all the users. Through this approach, the collusion of hosts and server can be prevented. The proposed communication protocol works as follows:

Consider three hosts *H1, H2 and H3*, where each host holds *itemLists I* as I_1, I_2, *and* I_3, respectively. Now each site wants to compute $I = I_1 + I_2 + I_3$ without revealing their local *itemLists* to each other. Each host computes shares of secret key as *share* $(I_1, H_1) = p_1(x)$, *share* $(I_2, H_2) = p_2(x)$, *share* $(I_3, H_3) = p_3(x)$. The last host interacting with server gets all the shares and adds all the received shares to compute $T(x) = p_1(x)+p_2(x)+p_3(x)$ and sends this result to the server. The server can decrypt the global *itemLists* and does not reveal individual users privacy.

Thus, the communication protocol 1 and communication protocol 2 are privacy-preserving for all users. Hence, we can say that user interest modeling is privacy-preserving for all users in *U*.

6.3 Privacy-preserving in the pseudo-user formation and delegation

A Pseudo-user communicates with the server to receive recommendations on behalf of real users. Real users, then calculate recommendations from pseudo-users to verify the importance of recommendation to them. No private information of the user is needed by the server to select a pseudo-user. Algorithm *SecurePseudoUSer* shows how pseudo-users are formed. One more crucial step is to delegate pseudo-users to interest groups. In this case, also no confidential information of pseudo-users is required to delegate them to the respective interest group. Thus, we can say that pseudo-user formation and delegation is privacy-preserving.

6.4 Privacy-preserving in the content recommendation

Content recommendations require association rule mining of *itemLists* received after secure distribution of users *itemLists* through protocol 1 and protocol 2. Privacy preservation feature of proposed protocols is discussed in section 4.6.4, which shows that user privacy can be protected against group members. All the encrypted *itemLists* from users are sent to the server via the host, and hence, the privacy of real users is protected from recommender server. After recommender server receives the *itemLists* of all the users in the system, the recommender server builds user-item rating matrix. However, the server cannot identify to which user the *itemLists* belong to. The server can generate recommendations using association rule mining approach.

Server-side recommendation [27] The server-side recommendation is only using the pseudo-users profiles to send recommendations. Real users calculate recommendation they are interested from pseudo-users. Therefore, we can say that the server is not aware of real users interest. Thus, it can be said that server-side recommendations do not exploit users privacy.

Client-side recommendation [27] The client-side recommendations are all computed based on the recommendation of pseudo-users profiles, just like in [27]. Pseudo-users have no user profiles stores to generate recommendations. The only interaction between pseudo-users and real users is, that real users calculate recommendations from pseudo-users. Therefore, client-side recommendations are also safe and privacy-preserving.

Overall, the privacy-preserving features of system model are explained in this chapter. Therefore, it can be said that the proposed system model which is based on [27] is privacy-preserving and especially protects users privacy with no loss of content information.

7. Evaluation

In this chapter, I use an experimental setup that is built to measure the performance and recommendation quality of the proposed system model. To equally evaluate recommendations and interest group analysis, we run the built prototype on the datasets: *Deezer* and *lastfm* which are online music streaming services. We perform evaluations to compare the accuracy of recommendations with different group sizes. We also analyze the recommendation quality compared with [27]. Then an evaluation is performed to choose the optimal number of interest groups. This system model on generating accurate recommendations to users participating in online social communities.

7.1 Experimental Setup

The proposed system model is tested in two parts. First, we analyze the recommendation accuracy of the proposed system model using experiments *precision (P)* and *recall (R)* [27]. We discuss the following experiments in detail. Then, we continue to analyze the computational costs of the proposed system model. The experiments are conducted on Windows OS 10 64 bit and 2.50 GHz—Core i5-4300U, 8GB CPU unit with JetBrains PyCharm Community Edition 2018.1.3.

The proposed model is tested using data from a popular online social community, *Deezer (Nov 2017)*[1] and *Lastfm*[2]. *Deezer* is a popular online music stream service. The dataset used in this work is from users in 3 Europen countries and is friendship network between the users. The dataset contains 85 distinct genres and 417,74 users. The other dataset used in this work is a popular online music service founded in 2002. *Lastfm* dataset used in this work is a subset of last.fm dataset. It contains 1227 users and 285 artists. In this work, we perform most of the evaluation on *Deezer*, as it is a friendship-based network.

7.2 Experiments *p* and *r*

Recommendation quality is calculated using *precision(p)* and *recall(r)* [27]. The equations of *p* and *r* are given as,

$$precision\ (u, r) = \frac{|Iu \cap Ir|}{|Ir|} \qquad (1),$$

$$Recall\ (u, r) = \frac{|Iu \cap Ir|}{|Iu|} \qquad (2)$$

Recommendation quality is defined as precision (u, r) and recall (u, r), Let Ir be the set of items recommended to users and the set of items a user u likes be Iu [29]. Here, *Precision* (u, r) defines the fraction of items a user u likes are actually recommended and *Recall* (u, g) defines the fraction of items liked by u are actually contained in his/her own likes [27][29]. Both *precision* and *recall* are required to measure the quality of the recommendation. If *precision(p)* and *recall(r)* is high, the quality of recommendations is also high[27].

[1] Deezer https://snap.stanford.edu/data/gemsec-Deezer.html
[2] Lastfm https://medium.com/radon-dev/item-item-collaborative-filtering-with-binary-or-unary-data-e8f0b465b2c3

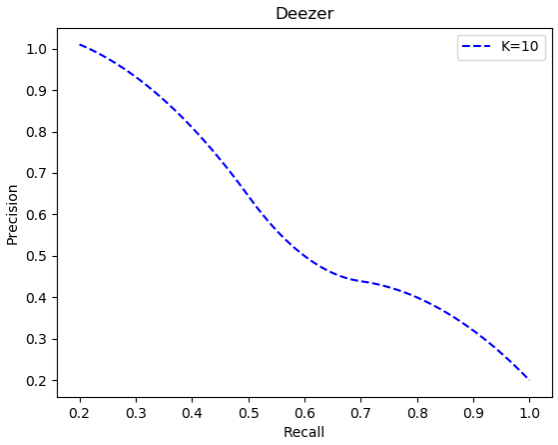
Example, from the dataset *Deezer*, user '*2222*' likes *items* 'Dance', 'Dancefloor' 'Electro' 'International Pop' 'Pop' 'Techno/House' and recommended items 'r' to user '*2222*' are 'Dance' 'Electro' 'International Pop' 'Pop' 'Alternative' 'Rap/Hip Hop' and so on. The *p* and *r* for the user '*2222*' can be calculated from the equations (1) and (2).

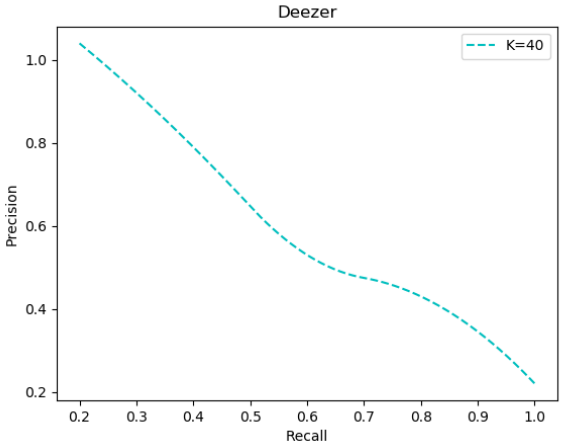
$$precision\ (2222, r) = \frac{|I2222 \cap Ir|}{|Ir|} = \frac{4}{5} = 0.8,$$

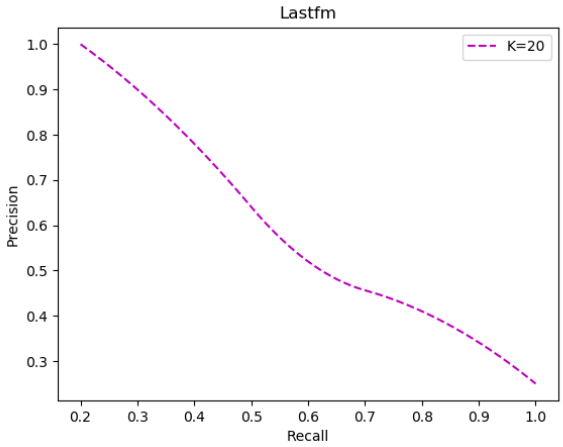
$$Recall\ (2222, r) = \frac{|I2222 \cap Ir|}{|Iu|} = \frac{4}{6} = 0.6$$

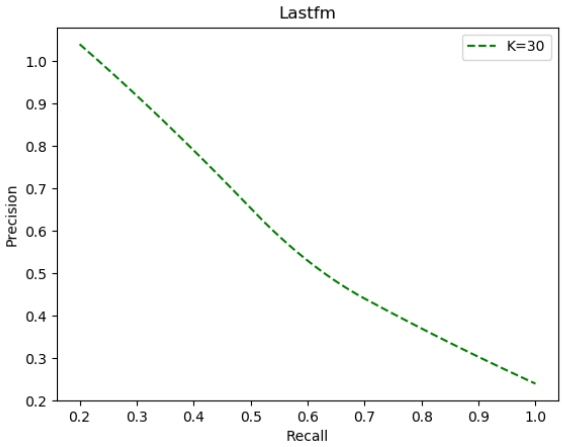
Therefore, user '*2222*' has the *precision* and *recall* values of 0.8 and 0.6 respectively.

Now, the experiments *p* and *r* are applied to the transposed dataset above with different k (user group size) values. Note: the user groups in the experiment are chosen randomly, a set of random users are selected to form user groups. The recommendation process is performed based on association rule mining approach. For the generation of association rules, we perform frequent itemset mining using Apriori algorithm as discussed in the previous section. Here, we specifically perform recommendation process on user group sizes of 10, 20, 30 and 40. Note that, experiments *p* and *r* are performed multiple times for each k value and the average values are taken to analyze the accuracy of recommendations generated.









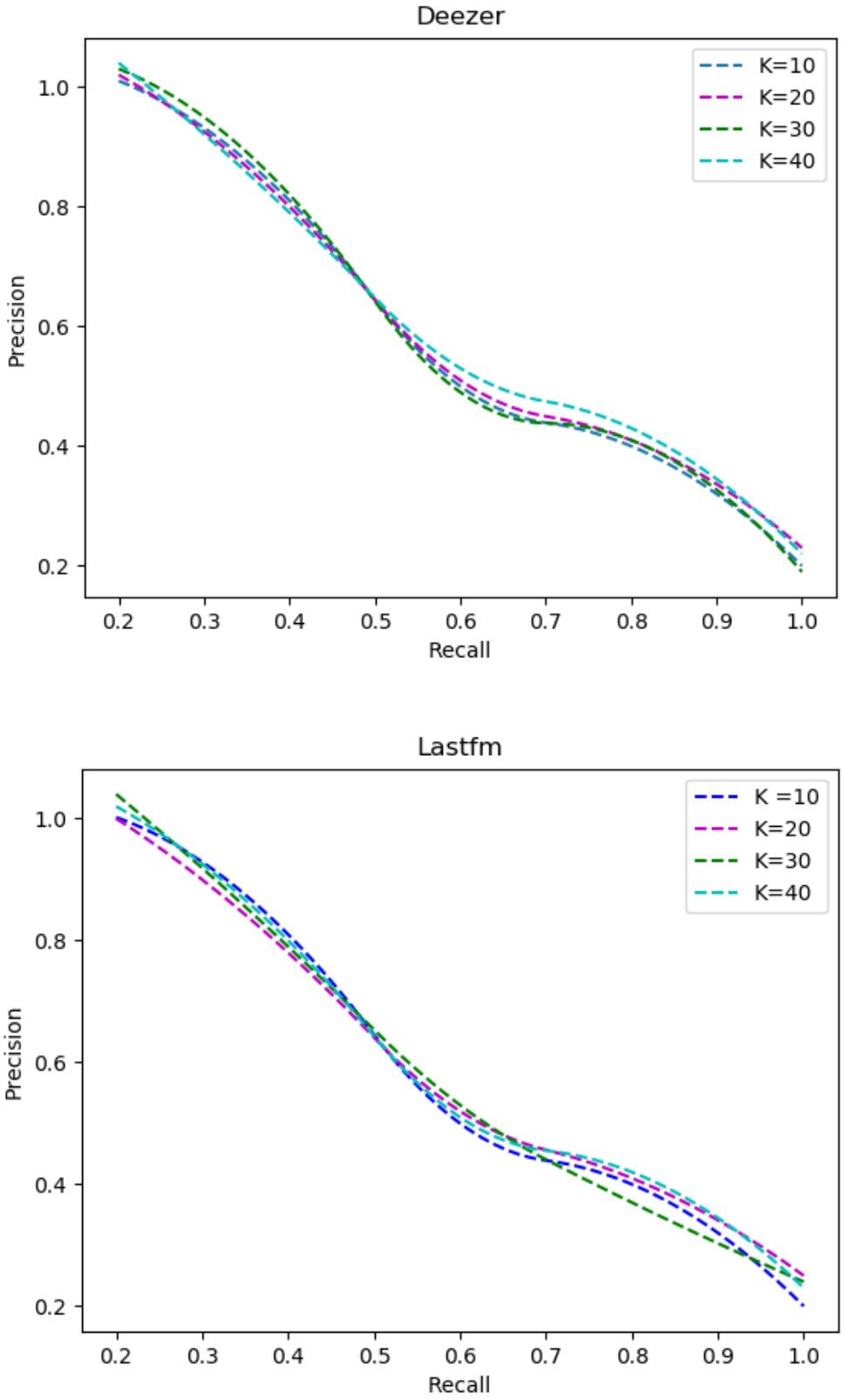


Figure 15. Recommendation quality of Deezer and Lastfm with group size (k=10 to 40)

From the figure 15, we can observe that the recommendation quality remains similar with the increase in k-value from 10 to 40. As the value of k increases, users are aggregated into the system. This often aggregates items to be recommended to users. As shown in YANA, the degradations are negligible with an increase in k-value. When compared with different k values, the degradation is less than 1%. This is calculated using Jaccard distance between the set values of *precision* and *recall* for different k values. Thus, it can be claimed that the proposed system model can generate a high-quality recommendation to users with different k values. i.e., with different user group size.

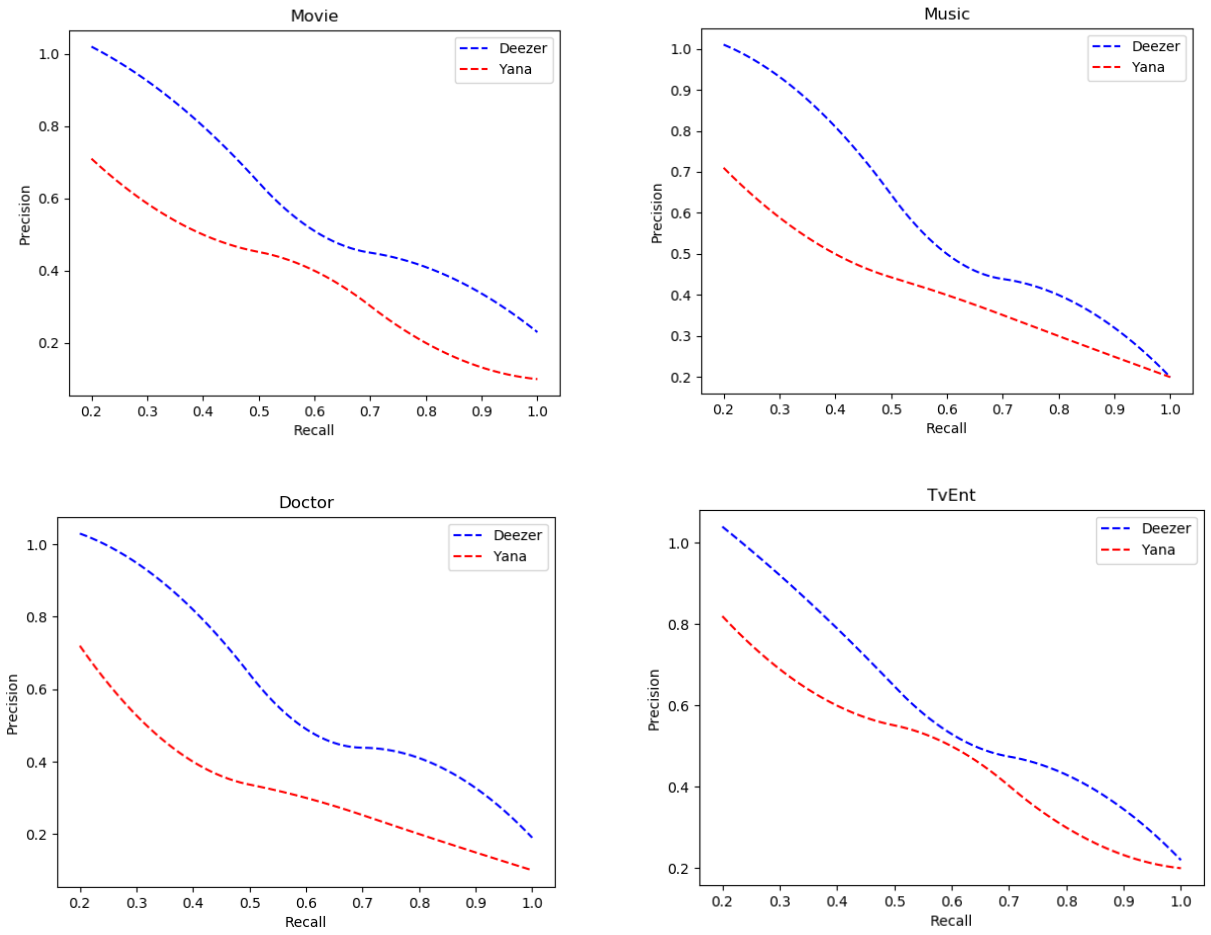


Figure 16. Recommendation quality of Deezer in comparison with different sub-communities of "YANA" [27]

Now, we compare the recommendation quality with "YANA" [27]. Recommendation quality in "YANA" are evaluated using data from "*Fudan BBS*[1]*"* [27]*." It is a popular online social community among Chinese universities and has over 60,000 users and 20,000 posts (every day)"*[27]. But, for our evaluation, we use *Deezer* and *Lastfm* dataset. Though a number of users and items vary in *Fudan BBS*, *Deezer*, and *Lastfm*. In this evaluation, we consider approximate recommendation accuracy of different sub-communities of *Fudan BBS* used in [27]. However, our evaluation shows that recommendation accuracy of our system outperformed in all the sub-communities of "YANA". The recommendation

[1] Fudan BBS, http://bbs.fudan.edu.cn.

qualities vary because “YANA” uses a collaborative-filtering method to make recommendations to real users [27]. The recommender server in “YANA” calculates recommendations based on pseudo-users similarity. While our approach uses association rule mining to generate recommendations [27]. Moreover, recommendations are not calculated based on pseudo-users preferences [27]. Items supporting the minimum support threshold are generated as recommendations and passed to pseudo-users. Real users calculate the recommendations from pseudo-users to know whether the recommendation is important to them or not. Therefore, considering the differences in the recommendation process, we claim that, this system model provides high recommendation accuracy when compared to “YANA”.

7.3 Interest group cluster analysis

In this experiment, we perform analysis of cluster similarity using Jaccard index [23] which is defined as intersection over union [23]. It is a statistical test applied for sets of clustered data to compare the similarity or diversity among the clustered sets.

$$JS\,(C, C') = \frac{|C \cap C'|}{|C \cup C'|} \qquad (9),$$

Let *c* and *c'* be the clusters, then the similarity or diversity of the *c* and *c'* is measured using equation (9). In our experiment, we perform analysis with different values of *c* and check the optimal number of clusters (k) for the dataset Deezer. As discussed in the previous section, we generate interest groups based on a k-centroid algorithm. It is necessary to choose an optimal number of clusters (k) to perform good intra-group similarity and generate accurate recommendations to users. After item-item similarity calculation of the dataset transaction. We choose k- centroids to cluster similar items into clusters knows as interest groups. Figure 17 below gives a brief idea of item-item similarity.

```
                    Deezer Item- item Similarity matrix

                     Acoustic Blues     ...      West Coast
  Acoustic Blues           1.000000     ...        0.000000
  African Music            0.044191     ...        0.005636
  Alternative              0.008042     ...        0.028171
  Alternative Country      0.037340     ...        0.011230
  Asian Music              0.000000     ...        0.004623
```

Figure 17. Item-item similarity calculation using Jaccard similarity in *Deezer*

```
                          Deezer – Interest groups

Interest group 0 = ['Comedy' 'R&B' 'Electro' 'Films/Games' 'Rock' 'Dancefloor'
'Techno/House' 'Dance' 'Rap/Hip Hop' 'Contemporary R&B' 'Film Scores' 'International
Pop' 'Alternative' 'Indie Rock' 'Disco' 'Dubstep']

Interest group 1 = ['Electro Hip Hop' 'Electro' 'Rap/Hip Hop' 'R&B' 'International
Pop' 'Dance' 'Techno/House' 'Contemporary R&B' 'Alternative' 'Dancefloor' 'Disco'
'Rock' 'Pop' 'Films/Games' 'Film Scores' 'Dubstep']
```

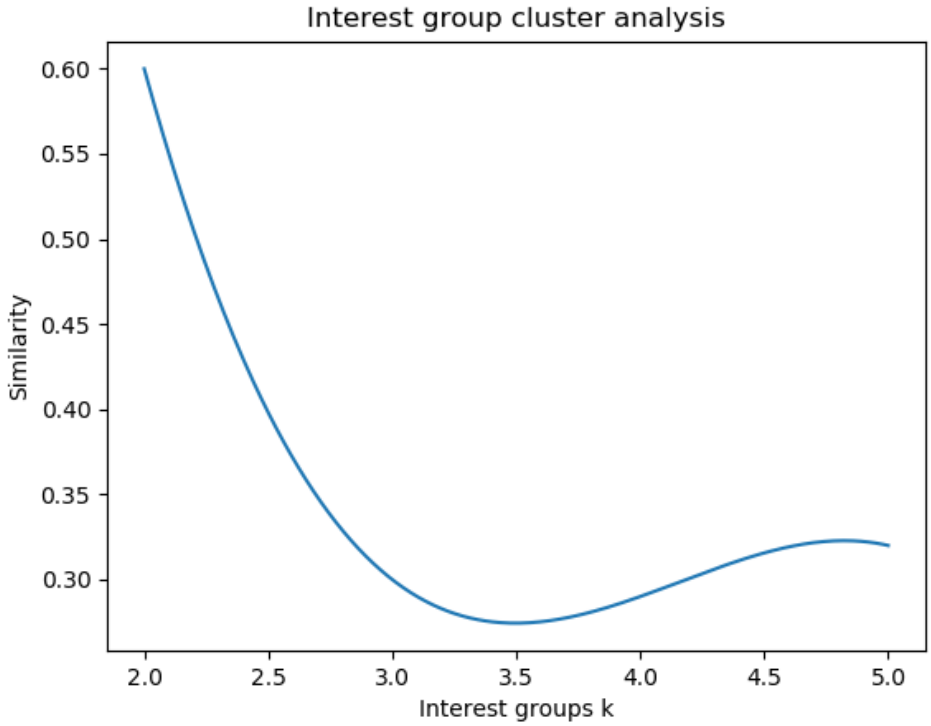
```
Interest group 2 = ['Jazz Hip Hop' 'East Coast' 'Soul & Funk' 'Old School' 'West
Coast' 'Dirty South' 'Grime' 'Contemporary Soul' 'R&B' 'Contemporary R&B' 'Rap/Hip
Hop' 'Jazz' 'Instrumental jazz' 'Comedy' 'Chill Out/Trip-Hop/Lounge' 'Rock']

Interest group 3 = ['Rap/Hip Hop' 'R&B' 'Electro' 'Dance' 'Pop' 'International Pop'
'Alternative' 'Contemporary R&B' 'Rock' 'Techno/House' 'Latin Music 'Films/Games'
'Film Scores' 'East Coast' 'Disco' 'Indie Rock']

Interest group 4 = ['Tropical' 'Norteño' 'Latin Music' 'Soundtracks' 'Reggae' 'Film
Scores' 'Films/Games' 'Contemporary Soul' 'Country' 'Brazilian Music' 'Indie
Pop/Folk' 'R&B' 'Contemporary R&B' 'Jazz' 'Indie Pop' 'Chill Out/Trip-Hop/Lounge']
```

Figure 18 Interest groups formed after computing k-centroid algorithm in *Deezer*

Figure 18 shows the interest group value with k = 5. Therefore, the interest groups formed are 5 (Interest group 0 - Interest group 4).



From figure 19, we observe that interest group cluster similarity is highest for k = 2. The measure of quality is defined by "goodness" [33]. A good clustering produces interest groups with high intra-group similarity and low inter-group similarity [27]. In figure 19, it can be seen that clusters formed with k =2 do not provide good inter-group similarity. Therefore the "goodness" quality of interest groups formed with k = 2 is low. However, for k = 3 and 4 the measure of similarity is low. i.e., 0.30 and 0.29 respectively. Hence, Interest groups formed with k-values 3 and 4 provide good interest groups with high intra-group similarity and low inter-group similarity in *Deezer* and similarly, optimal interest group in *lastfm* dataset is between k = 4 to 5.

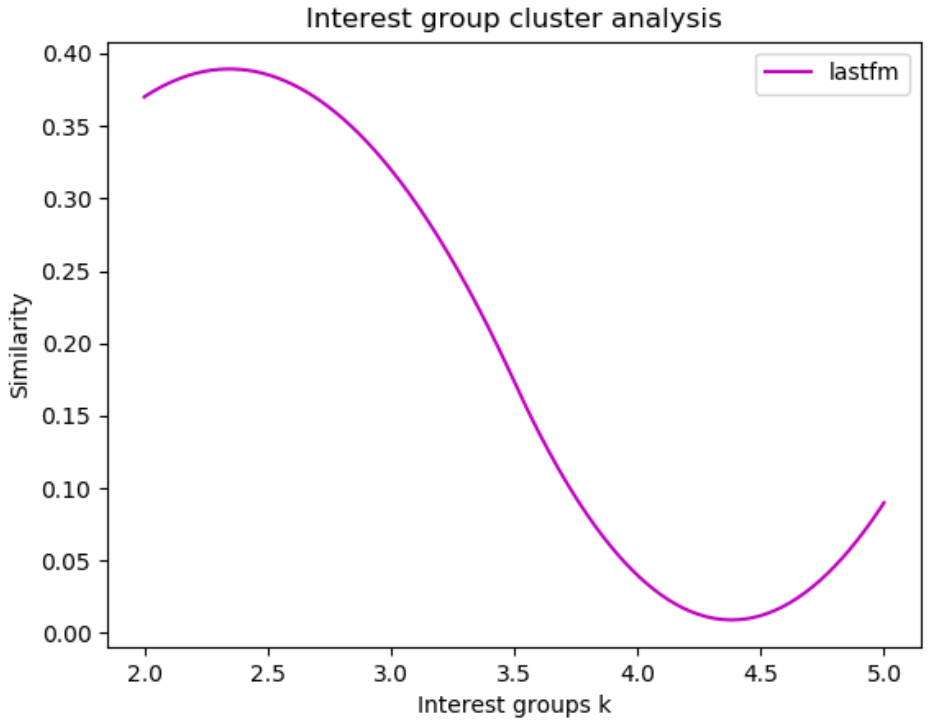


Figure 19. Analysis of optimal interest groups for *Deezer* and *lastfm* datasets

Therefore, we can conclude that the system model provides high-quality recommendations irrespective of user group size. We also perform cluster analysis and found an optimal number of clusters for the datasets *Deezer* and *lastfm*.

8. Conclusion and future work

8.1 Conclusion

In recent years, there have been many recommender systems to generate recommendations to the users in online social communities. The growth of online social communities has led to various privacy concerns. In online social communities, recommender systems make use of users sensitive information to make personalized recommendations to users. There is a need to protect users information from recommender server. In this thesis, a privacy-preserving content-based recommender system for online social communities is proposed, which is similar to "*YANA – an efficient privacy-preserving recommender system for online social communities*" [27]. The system model designed organizes users with diverse interests into user groups. User groups protect the privacy of users from recommender server using two protocols, elliptic curve cryptography, and Shamir's secret sharing scheme. Within each user group, interest groups are created which clusters similar items into clusters. Interest groups help to generate pseudo-users, who are responsible to receive recommendations from the recommender server. Real users inside the user group calculate personalized recommendations from pseudo-users. We have evaluated the system model on two popular real-world datasets *Deezer* and *Lastfm*, to check the quality of recommendations and perform interest group analysis to find an optimal number of interest groups. Our evaluation results show that this system model achieves high recommendation quality irrespective of number users in a given user group while protecting users privacy.

8.2 Future work

Like any research, there are a lot of improvements and that can be applied to this work. We discuss the improvements or additions that can be applied to this work.

- Due to time constraint and unavailability of several usage constraints, we could not perform evaluation using protocols mentioned in chapter 5 section 5.3. The protocols are elliptic curve cryptography and Shamir's secret sharing scheme. This helps to evaluate the efficiency of the system model in terms of computation cost.
- The system model should be implemented on devices, using real-time networks to evaluate the latencies of generating recommendations to users.
- In recent years, due to various advancements in natural language processing approaches, clustering of items can be performed the above-mentioned approaches. Clustering of items is crucial as this helps to generate accurate recommendations to users in a given group.

Appendices

Bibliography

[1] C. A. Yeung, I. Liccardi, K. Lu, O. Seneviratne, and T. Berners-Lee, "Decentralization : The Future of Online Social Networking," *W3C Work. Futur. Soc. Netw. Position Pap.*, vol. 2, pp. 1–5, 2009.

[2] A. De Salve, P. Mori, and L. Ricci, "A survey on privacy in decentralized online social networks," *Comput. Sci. Rev.*, vol. 27, no. June, pp. 154–176, 2018.

[3] T. Paul, S. Buchegger, and T. Strufe, "Trustworthy Internet," no. March, 2011.

[4] S. Badsha, X. Yi, and I. Khalil, "A Practical Privacy-Preserving Recommender System," *Data Sci. Eng.*, vol. 1, no. 3, pp. 161–177, 2016.

[5] P. Katsoulakos, M. Koutsodimou, A. Matraga, and L. Williams, "A CSR oriented Business Management Framework Part A - CSR Foundations - Historic perspective of the CSR movement," *CSR Quest Sustain. Framew.*, pp. 1–19, 2004.

[6] R. Burke, "Hybrid Recommender Systems : Survey and," no. C, pp. 2005–2008, 2008.

[7] J. Danasana, R. Kumar, and D. Dey, "MINING ASSOCIATION RULE FOR HORIZONTALLY PARTITIONED DATABASES USING CK SECURE SUM," vol. 3, no. 6, pp. 149–157, 2012.

[8] N. Jain and P. A. Singh, "a Survey on Privacy Preserving Mining Various Techniques With Attacks," *Int. J. Res. Comput. Appl. Robot. www.ijrcar.com*, vol. 5, no. 5, pp. 16–21, 2017.

[9] M. Kaur and S. Kang, "Market Basket Analysis : Identify the changing trends of market data using association rule mining," *Procedia - Procedia Comput. Sci.*, vol. 85, no. Cms, pp. 78–85, 2016.

[10] M. S. S. Borhade and S. V Gumaste, "Defining Privacy for Data Mining- An Overview," vol. 5, no. 6, pp. 182–184, 2015.

[11] C. C. Aggarwal and P. S. Yu, "A General Survey of Privacy-Preserving Data Mining Models and Algorithms," pp. 11–52, 2008.

[12] M. Shridhar and M. Parmar, "Survey on Association Rule Mining and Its Approaches," no. 3, pp. 129–135, 2017.

[13] H. Chahar, B. N. Keshavamurthy, and C. Modi, "Privacy-preserving distributed mining of association rules using Elliptic-curve cryptosystem and Shamir's secret sharing scheme," *Sadhana - Acad. Proc. Eng. Sci.*, vol. 42, no. 12, pp. 1997–2007, 2017.

[14] P. Chen, "Secure Multiparty Computation for Privacy Preserving Data Mining."

[15] M. Y. Malik, "Efficient Implementation of Elliptic Curve Cryptography Using Low-power Digital Signal Processor," no. x, pp. 1464–1468, 2010.

[16] M. O. F. Success, "International Achievements of Polish Urban Planning," pp. 15–26.

[17] R. Kumar and A. Anil, "Implementation of Elliptical Curve Cryptography," vol. 8, no. 4, pp. 544–549, 2011.

[18] S. A. Chaudhry, M. S. Farash, H. Naqvi, and M. Sher, "A secure and efficient authenticated encryption for electronic payment systems using elliptic curve cryptography," *Electron. Commer. Res.*, no. June, 2015.

[19] J. Irani, "Clustering Techniques and the Similarity Measures used in Clustering : A Survey," vol. 134, no. 7, pp. 9–14, 2016.

[20] G. J. Torres, R. B. Basnet, A. H. Sung, S. Mukkamala, and B. M. Ribeiro, "A Similarity Measure for Clustering and Its Applications," 2008.

[21] S. Wagner and D. Wagner, "Comparing Clusterings - An Overview," no. 001907, pp. 1–19, 2007.

[22] J. Soler, F. Tenc, and L. Gaubert, "Data Clustering and Similarity," pp. 492–495.

[23] L. Zahrotun, "Comparison Jaccard similarity, Cosine Similarity and Combined Both of the Data Clustering With Shared Nearest Neighbor Method," *Comput. Eng. Appl.*, vol. 5, no. 11, pp. 2252–

4274, 2016.
[24] C. Chang, J. Yeh, and Y. Li, “Privacy-Preserving Mining of Association Rules on,” vol. 6, no. 11, pp. 259–266, 2006.
[25] M. Kantarcioglu and C. Clifton, “Privacy-preserving distributed mining of association rules on horizontally partitioned data,” *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 9, pp. 1026–1037, 2004.
[26] شماره دوم 1376 سال ایران شناسی جامعه جامعه مجله“ ,ایران شناسی جامعه .م,” vol. 16, no. 9, p. 27, 2004.
[27] D. Li, Q. Lv, L. Shang, and N. Gu, “Efficient privacy-preserving content recommendation for online social communities,” *Neurocomputing*, vol. 219, no. October 2016, pp. 440–454, 2017.
[28] S. Badsha, X. Yi, I. Khalil, and E. Bertino, “Privacy Preserving User-Based Recommender System,” *Proc. - Int. Conf. Distrib. Comput. Syst.*, no. August, pp. 1074–1083, 2017.
[29] D. Li, Q. Lv, H. Xia, L. Shang, T. Lu, and N. Gu, “Pistis: A privacy-preserving content recommender system for online social communities,” *Proc. - 2011 IEEE/WIC/ACM Int. Conf. Web Intell. WI 2011*, vol. 1, pp. 79–86, 2011.
[30] J. MacQueen, “Some Methods for classification and Analysis of Multivariate Observations,” *5th Berkeley Symp. Math. Stat. Probab. 1967*, vol. 1, no. 14, pp. 281–297, 1967.
[31] T. N. Jabeen and M. Chidambaram, “Privacy Preserving Association Rule Mining in Distributed Environments using Fp-Growth Algorithm and Elliptic Curve Cryptography,” *Indian J. Sci. Technol.*, vol. 9, no. 48, 2017.
[32] A. Shamir, “Shamir - 1979 - How to share a secret.pdf,” pp. 612–613, 1979.
[33] H. Finch, “Comparison of Distance Measures in Cluster Analysis with Dichotomous Data,” vol. 3, pp. 85–100, 2005.