Table of Contents

2. Theoretical Foundation

Recommender systems

A recommender system provides a meaningful and useful set of recommendations to users based on information about user preferences [1]. The information gained from users can be derived explicitly or implicitly. Information used in recommender systems can be briefly categorized as [2]:

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 purchased. Positive, negative, or something more descriptive is few feedbacks provided by users. Therefore, is explicit information. For example, the review given to a product purchased in Amazon.
3) User preferences can be explicitly rated items on a scale of 1-5 stars or maintaining 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. These preferences can also predict other items that might also be of interest to the user in the future [1]. 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 [2]. Here, I briefly give an overview of proposed recommender systems above:

Collaborative Filtering (CF) [1] [2] is a widely used recommender system. In CF, content items are rated by each user. These ratings determine 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 [2] 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) [2]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 [2] 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 [3].

Distributed Privacy-preserving

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. Participants may wish to perform computations collaboratively to obtain aggregate results but 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 [4][5]. 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[4][5]. 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[5][4]. In a hybrid distribution of data, data are distributed either first horizontally and then vertically or vice-versa.

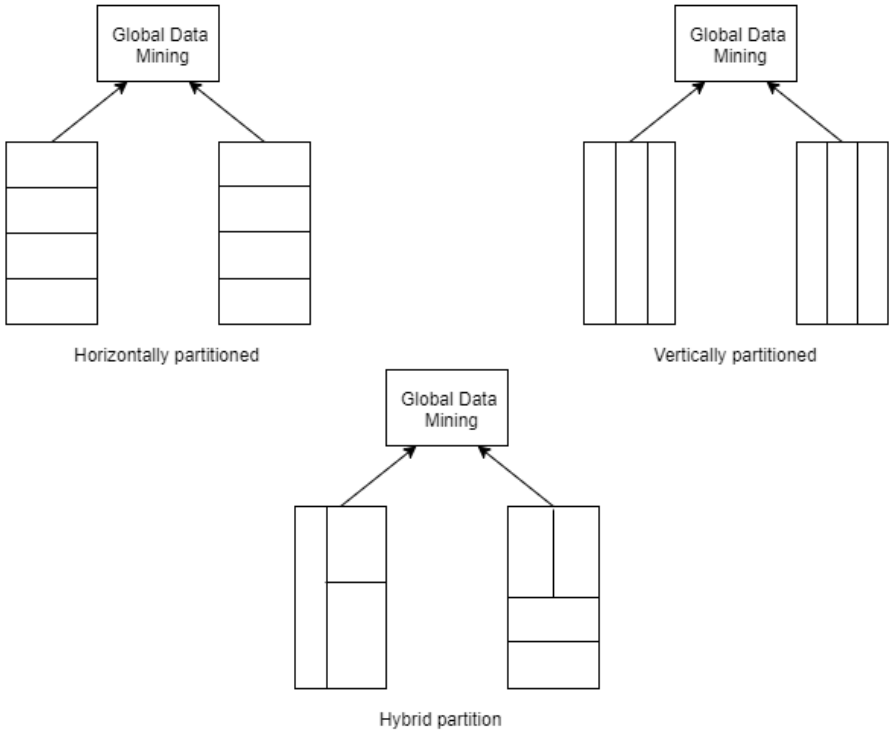


Figure 1 Classification of Data distribution

Several data mining techniques and algorithms are available to discover meaningful patterns and rules. These techniques have been discussed briefly in table 1 [6].

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. Data mining techniques

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. Database DB = $\{t_1, t_2, t_3,...,t_m\}$ is a set of size m transactions. In this, each transaction t is called an itemset if $t \subseteq I$ [4]. For an itemset $X \subseteq I$, a transaction t contains X if and only if $X \subseteq t$. An association rule is an implication $X=>Y$ where $X \subseteq I$, $Y \subseteq I$ and $X \cap Y = \emptyset$ [4]. The support value of an association rule $X \subseteq Y$ can be derived as follows [4]. The rule has support value S if a fraction of transactions that contain both X and Y is S[4] . The confidence of this rule is C is the measure of how often items in Y appear in database DB that containing X and Y is C[4].

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

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

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[7]. Agarwal proposed frequent pattern mining for market basket analysis and association rule mining[7]. The primary frequent pattern algorithms can be classified into two ways [7]:

1) Candidate generation approach.
   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 item set and therefore the name Apriori. This algorithm works on iterative approach or level wise approach[7]:

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[8]. Algorithm 1[9], 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 [7], 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

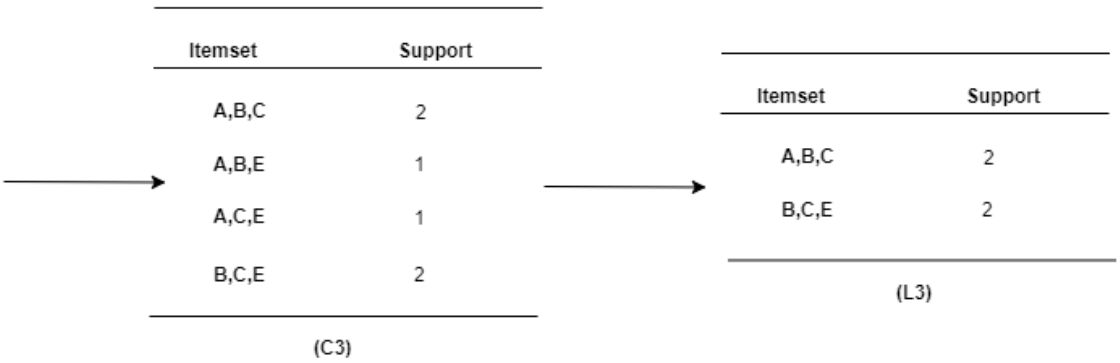
(C2)

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.



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

(C3)

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

(L3)

Table 5. Candidate generation 3 and Large-itemsets 3

Elliptic Curve Cryptography (ECC)

ECC is a public key cryptography system[10]. In this type of cryptography system each user or a device participating have a public key and a private key[11]. Also, each user or a device performs cryptographic operations associated with the keys [11]. 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[11][10]. ECC is known to be an efficient cryptographic scheme compared to earlier cryptographic key such as RSA, DSA and DH[12] [13] [11]. For example, RSA uses large numbers for its operation. Therefore, larger would be the numbers in case of more security[10]. Basics of ECC is explained below [10]:

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 [10]:

1) Prime finite fields and
2) Binary finite fields

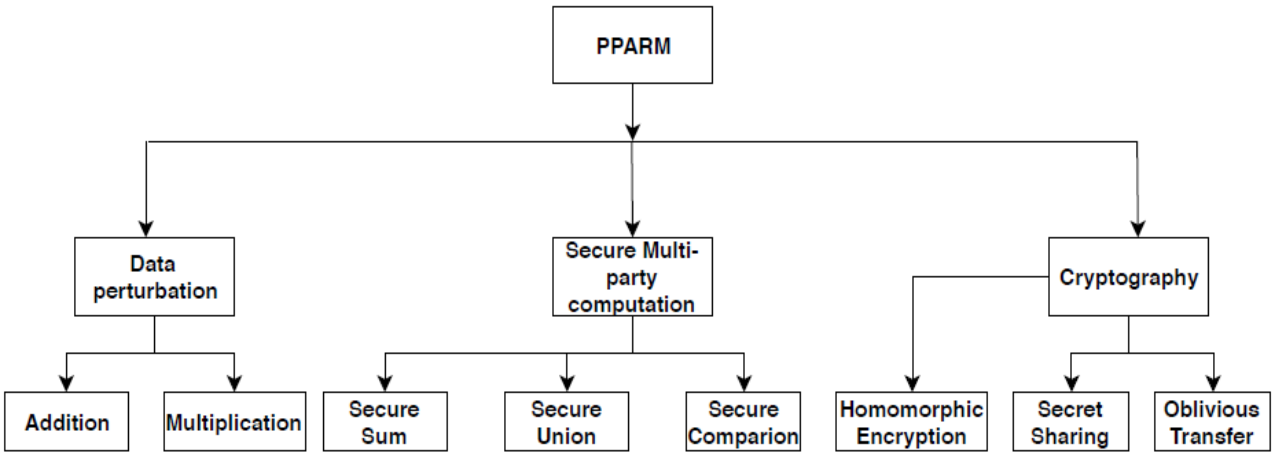
Advantages of ECC [10]:

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.

Shamir's secret sharing scheme

3. State-of-Art

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 secure union, secure comparison and secure sum; and cryptography approaches, which are divided into Shamir's secret sharing, oblivious transfer and homomorphic encryption (see figure 2).



Data perturbation techniques [1–3] 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

Dongsheng Li, Qin Lv [14] proposed YANA ("you are not alone"), 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 set of pseudo-users are generated for each user group. A unique interest is delegated to each pseudo-user generated and therefore, pseudo-users covers all interests in given a user group [14]. Recommender server interacts with real users through pseudo-users. Personalized recommendations are calculated on users side after receiving recommendations from the server[14]. Thus users, private data is not exposed to the server. To ensure user privacy the authors in [14] 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 [14] "$Pr_{host}(u) = K_u/|U|$". 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 formation of interest groups, each user in a given group will have an interest distribution over these interest groups based on their liked items and to which interest groups those items belong to [14]. 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. The data structure *HashVector*⟨ key, value ⟩ 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 divides his/her input value into *n* parts such that the sum of the *n* parts equals the input value [14]. Users obfuscate the values by sharing the values between themselves and send the sum of their local obfuscated parts to the "host", and then "host" computes and returns the sum back 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.

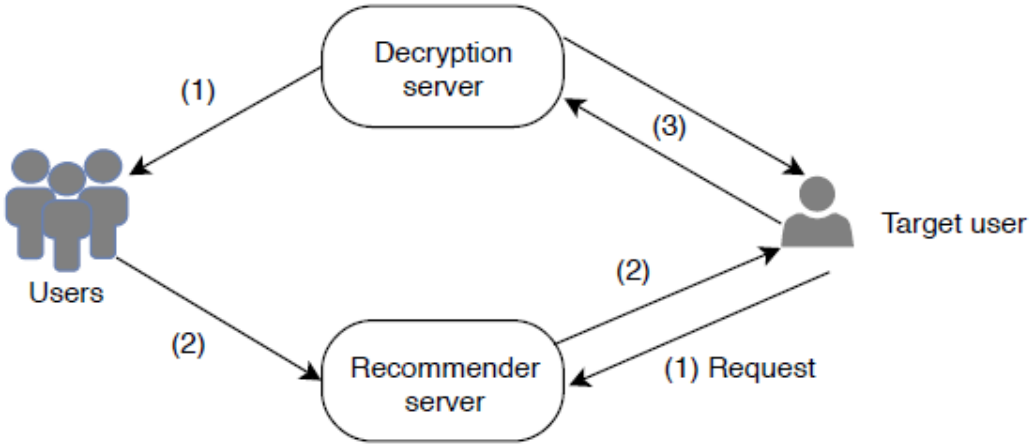
In the fourth protocol, *SecureRate* algorithm is proposed as no user would like to expose his/her interests, a privacy-preserving protocol is needed to maintain an interest profile for pseudo-users and item ratings for each pseudo-user. Users in a given group run the *SecureSum* protocol as discussed in *SecureSearch*. This protocol creates pseudo-user profiles which are required by the server to make recommendations.

In 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.

Shahriar Badsha, Xun Yi [15] 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:

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:

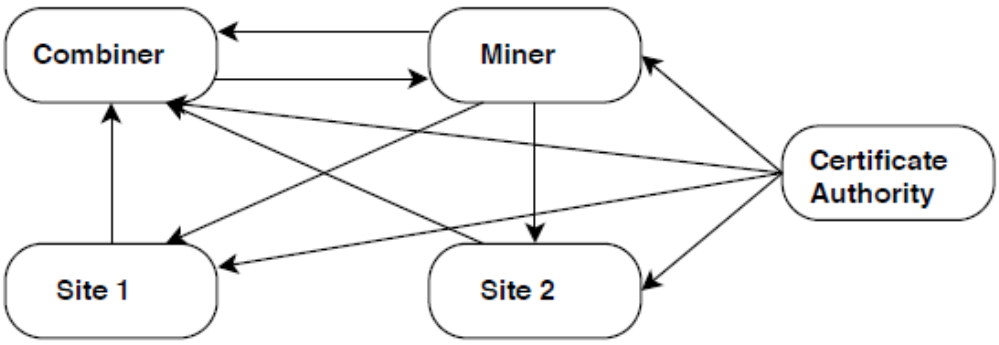
1) A user is also known as target user participates in the recommendation process by sending a request to RS. Encrypted ratings of the other users are received by the target user via the RS and locally determines the similarity in encrypted domain. The resultant ciphertexts are returned to the RS.
2) 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 [1] 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 ciphertexts 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. The server computes similarity among the items securely as discussed above and allows users to 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.

Chahar, Harendra Keshavamurthy, B. N. Modi, Chirag [8] 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 requires 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 the figure , consider 4 sites $Site_1$, $Site_2$, $Site_3$ and $Site_4$ containing 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. Then the protocol works as follows,

1) Each site generates local maximum frequent itemsets (MFI) and sends encrypted local MFI to the miner.
2) 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.
3) 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.
4) Each site sends local database size in a similar way as local support of itemset.
5) 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 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

Murat and Chris [16] 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 two phases in this approach include,

1) Finding frequent candidate itemsets on one or more sites and,
2) To verify those candidate itemsets meet the global support/confidence thresholds.

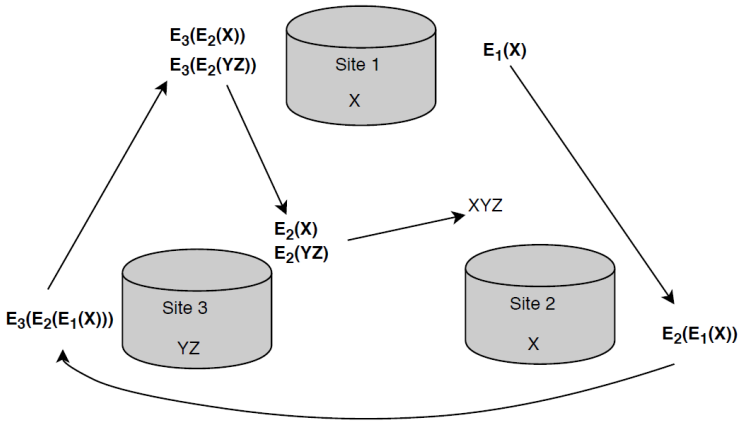


Figure 2 - Determining global candidate itemsets

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. Each site encrypts all itemsets that it receives, and this process is continued until all the sites have encrypted all itemsets. 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).

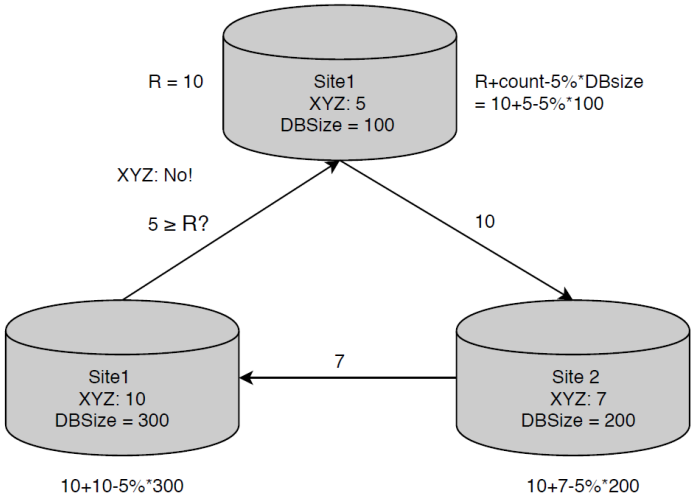


Figure 3. Determining if itemset support exceeds 5 percent threshold

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. Each site computes their local support. In the figure, the itemset XYZ is known to hold support count at one or more sites. In this process, the first site chooses a random value *R* and adds to *R* the amount by which its support for XYZ exceeds the minimum support threshold. 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 resulting value is tested to see if it exceeds the Random value using a secure comparison. If so, itemset XYZ is supported globally. It is a not a collusion-resistant protocol.

4. Methodology

In this chapter, an efficient privacy-preserving recommender system for users in online social communities is designed. This system model 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. 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 [14] 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. Each pseudo user delegates a unique interest group, and the union of all pseudo-users covers all interests of users in a user group [14]. 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 operations[17]. Consider an online social community and its associated recommender server, the following operations are performed by any user in the online community[17]:

- Post or comment on an item shared or recommended by other users
- Read the content item posted or commented on by other users and
- Finally, request recommendations from recommender server

From the above-mentioned operations, massive and diverse online content is generated by the users. The close interactions between the users have raised new concerns on recommender systems, among which user interest privacy preserving is a key challenge that needs to be addressed. Further, public and private information of users in the online social community can be differentiated [17]. 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[14] [17].

Given a user *u* and an online post (item) *i*, if *u* has posted/read/commented on *i*, we say u is interested in *i*, then we denote *u's* rating to *i* as $r_{i,u}$ = 1[14][17]. Otherwise, $r_{i,u}$ = 0[14][17]. Based on the binary ratings (“0” or “1”) [14], 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 [14]. For instance, 1–5 ratings can be normalized by dividing the values by 5, so that “1, 2, 3, 4, 5″ will be normalized to “0.2, 0.4, 0.6, 0.8, 1.0″, respectively. We normalize 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 frequently 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 [14]. However, these techniques do not help recommender systems to achieve accurate recommendations to the users [14]. So, a privacy-preserving recommender system is required to achieve accurate recommendations. To protect individual user's privacy, a high-level system design is proposed which is similar to [14]. 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 accurate recommendations.

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 identify the true interests of users. Interest group identification ensures that users receive no “uninterested” items during the recommendation process. In this system model, a k-centroid clustering algorithm is adopted to identify 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. The recommendations made to the pseudo-users are used by real users to calculate their own personalized recommendations.

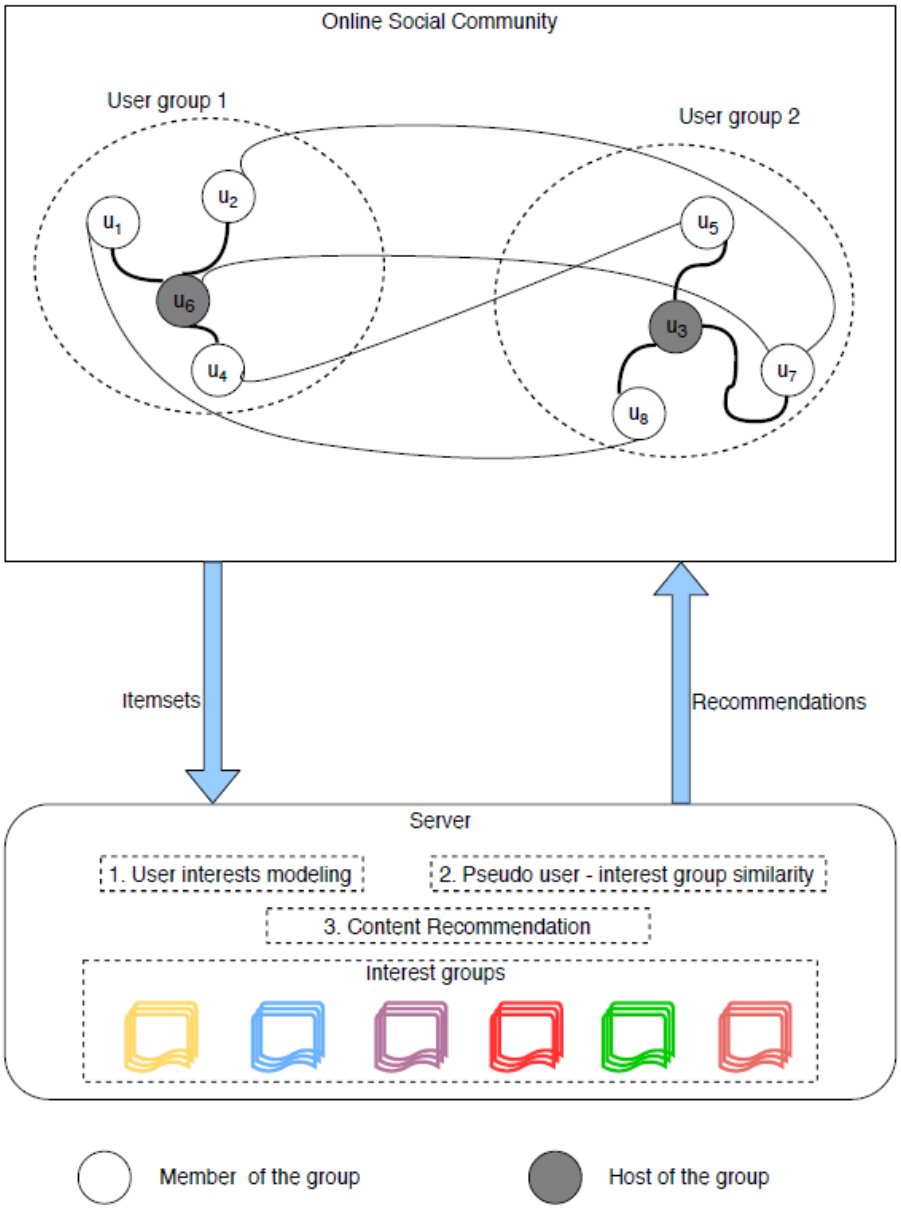


Figure 1 System model of a privacy-preserving content recommender for online social communities

4.2 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 transmit *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[14]. p_g is a set of pseudo-users who interact with the server on behalf of the real users in g to deliver recommendations[14].

User groups can be formed in privacy-preserving fashion to hide the contents of each user from a set of users participating in the same group. A group construction protocol is proposed which can automatically organize users into groups with diverse interests in a distributed and privacy-preserving fashion [14]. User group construction is shown in "*Algorithm 1*" [14] and for each user group *g* constructed, S_u is the number of users in a group and should be no less than 3. If *g* only contains two users (u_1 and u_2), then u_1*'s* privacy could be easily inferred from their joint computation results by u_2, and vice versa[14]. For a user group g with $S \geq 3$, users' privacy can be protected in *g*. The formal proof is shown in[14]. 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[14]. Thus, once a user chooses to leave a group, the other users in the same user group should check whether the requirement $S \geq 3$, i.e. size of g is greater than or equal to 3 [14]. The user group does need not to be changed if the requirement is met else, they should re-construct new user group using the *SecureGrouping* algorithm by dismissing the previous group [14].

Algorithm 2. SecureGrouping(U).

Require: *U* is the set of users in the Social community.
1: **For** each user $u \in U$, *Su* is the expected user group size of *u*;
2: **while** Not all users in *U* are in user groups **do**
3: **for** each $u \in U$ who has not joined any user group **do**
4: u chooses to be the "host" of a user group with probability $Pr_{host}(u) = S_u/|U|$
5: **if** *u* is host then
6: *u* invites its friends to join its group;
7: **end if**
8: **end for**
9: **for** each **do**
10: **Let** *Hu* be the set of u's friends who are hosts of user groups;
11: **if** $Hu \neq \emptyset$ **then**
12: u randomly chooses $v \in Hu$ and joins the group of *v*;
13: **else**
14: **Let** *Ju* be the set of u's friends who already joined the user group;
15: **if** $Ju \neq \emptyset$ then
16: *u* randomly chooses $v \in Ju$ and joins the group of *v*;
17: **end if**
18: **end if**
19: **end for**
20: **for** each user group *g* **do**
21: **Let** *u* be the host of *g*;
22**:** **if** $|u_g| \geq S_u$ **then**
23: *g* is formed;
24: **end if**
25: **end for**

26: **end while**

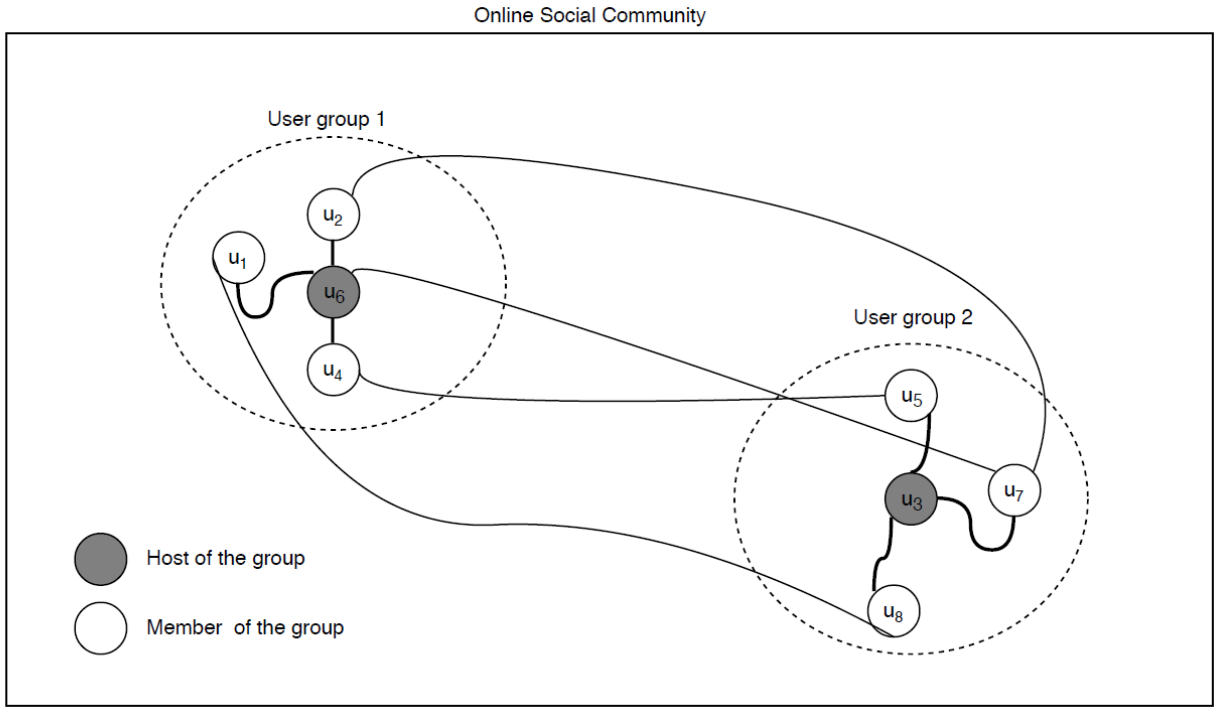


Figure 5 - User grouping in online social communities

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 [14]. 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.

4.3 Users Interest modeling: Interest groups definition and construction

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

Definition 2: A set of interest groups $I_g = \{I_{g1}, I_{g2}, I_{g3}....I_{gk}\}$, where k is the number of interest groups, in which $I_{gk} = \{i_1, i_2, i_3, ..., i_m\}$ is a set of items and c_g belongs to I_g is the centre 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 $i \neq j$, $I_{gi} \cap I_{gj} = \emptyset$ [17].

In this model, user interest modeling is performed by privacy-preserving user interest clustering algorithm, which clusters similar items into interest groups. After interest group modeling, each user group will have interest groups distribution to generate pseudo-users. Here k-centroid clustering method

is adopted to cluster similar items [14] [18]. The workflow of k-centroid clustering approach is described in Figure 3,

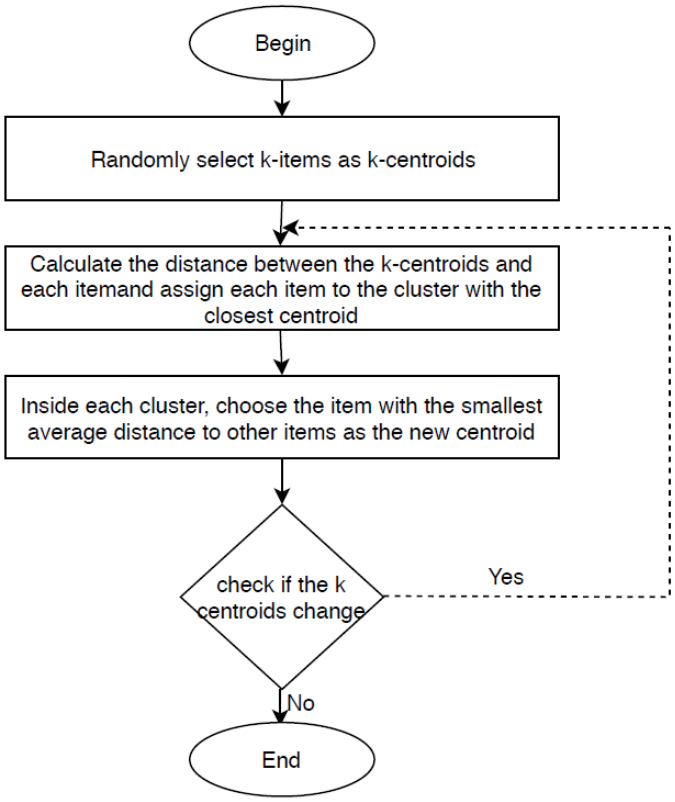


Figure 6 - Workflow of the k-centroid algorithm

Challenges in identifying interest groups [14] [17]:

- 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.
- Privacy-preserving item similarity computation.

4.3.1. The privacy-preserving item distance calculation

Another challenge in the k-centroids clustering process is to compute the distance between items (required for similarity computation) efficiently without compromising user privacy. To preserve user privacy, two communication protocols are proposed. The first protocol is based on Elliptic curve cryptography [8] and the second one is based on Shamir's secret sharing scheme[8]. 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 [14] [19]. It has the following property,

$$Jaccard\ similarity\ (i_1, i_2) = \frac{|i1 \cap i2|}{|i1 U\ i2|} \qquad (1),$$

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

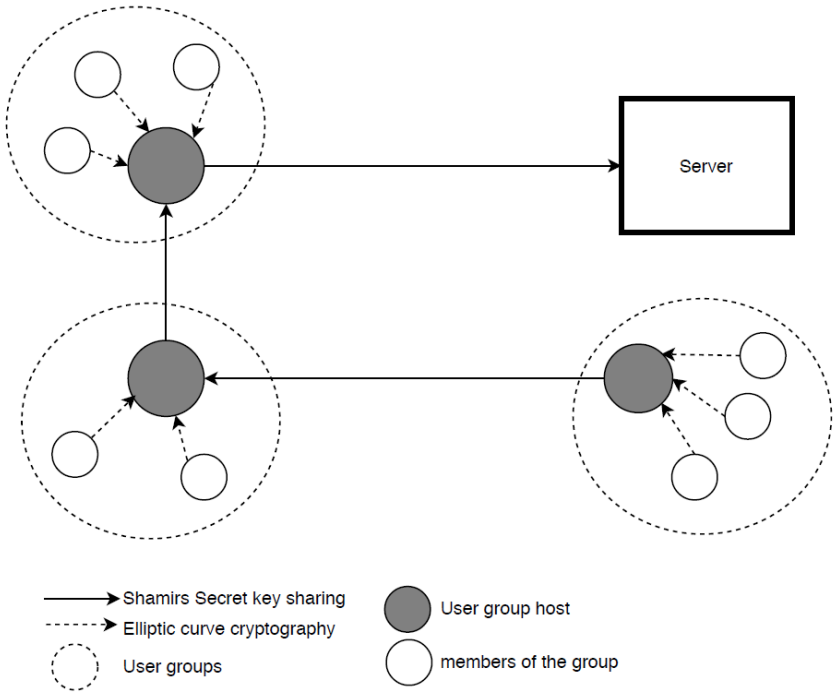


Figure 7 - Proposed communication protocol

Phase 1 proposed protocol based on Elliptic-curve cryptography[8][12]
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 [11] [8] [20]. The messages are signed .i.e. encrypted with the help of secret key of Elliptic-curve-based Paillier public key cryptosystem before sending it to hosts of the group. This helps in validating the integrity and authenticity of a message [8].

- 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 [8] [21]
To prevent the collusion between host and server, Shamir's secret sharing scheme 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 which are 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. each host generates a polynomial of degree *K*. 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 $k = n\text{-}1$. 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 [14]. The interest groups formed can help pseudo-users formation inside each user group and also helps In recommendation accuracy[14].

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 $I_{g2} = \{i_3, i_5, i_6, i_8, i_9\}$.

4.4 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 user acts as a "delegate" for an interest group, and the recommendations that the server makes to the pseudo-user can be utilized by real users to calculate personalized recommendations.

4.4.1. Pseudo-users formation

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, illustrates how pseudo-users are generated in a given user group.

4.4.2. The Delegation of pseudo-users to interest groups

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: $P_s = \emptyset$;
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 a user 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: $P_s = \{u\}$;
12: **end for**
13: **end for**
14**: return:** P_s;

4.5 Content Recommendations

After user grouping and generation of pseudo-users in each user group, the server can collect the *itemLists* of all users which is achieved from the previous section. The server can make recommendations to the pseudo-users based on the association rule mining approach.

4.5.1 Server-side recommendation Server requires standard user-item rating matrix, which is required in association rule mining to generate association rules which are considered as recommendations. From the section 4.3.1, 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.

For the recommender server, Apriori algorithm is employed for discovering association rules to recommend items to users. In this system, the server cannot see real user data, but only global user data which is privacy-preserving as no real user data is revealed, similar to [14]. 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[6]. 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 sets of items called itemsets, T be transaction that contains a set of items such that $T \subseteq I$, 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.

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 1. User-item rating matrix

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 1, shows the user-item matrix where '1' denotes that user likes the item and '0' denotes user is not interested in an 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. The server receives the support of all the items that occur in all the transactions.

Step1: Server then creates a candidate itemsets table for all the items along with the support count. In this case, Table 2. Shows candidate itemsets C_1 that have been generated. The transaction count in table 1. 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 2. Candidate itemsets C_1

Step 2: Only those elements are important for which the support value is greater than or equal to the 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 are eliminated

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

Table.3 Large-1 L_1 itemsets

Step 3: In the next step, all the possible pairs of the significant items are generated keeping in mind that the order doesn't matter, i.e., itemset XY is same as itemset YX. To generate all the possible pairs, take the first item and pair it with all the others 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

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.

Table 4. Large-2 L_2 itemsets

Therefore, the L_2 itemsets generated are shown in table 4.

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. And, possible 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 5. Association rules for recommendation

4.5.2 Client-side recommendations The item recommendation from the server are 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. On the other hand, if a real user does not have the antecedent in his itemList, the recommendation is not received. 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 receives the recommendation. Otherwise, the recommendation is not received by a real user.

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, *R1* and *R2* are received by all users in group 1 as $r_i > 0$, while *R3* is not received by any user in group 1 as $r_i = 0$. In group 2, *R1*, *R2*and *R3* are not received by any user as $r_i = 0$.

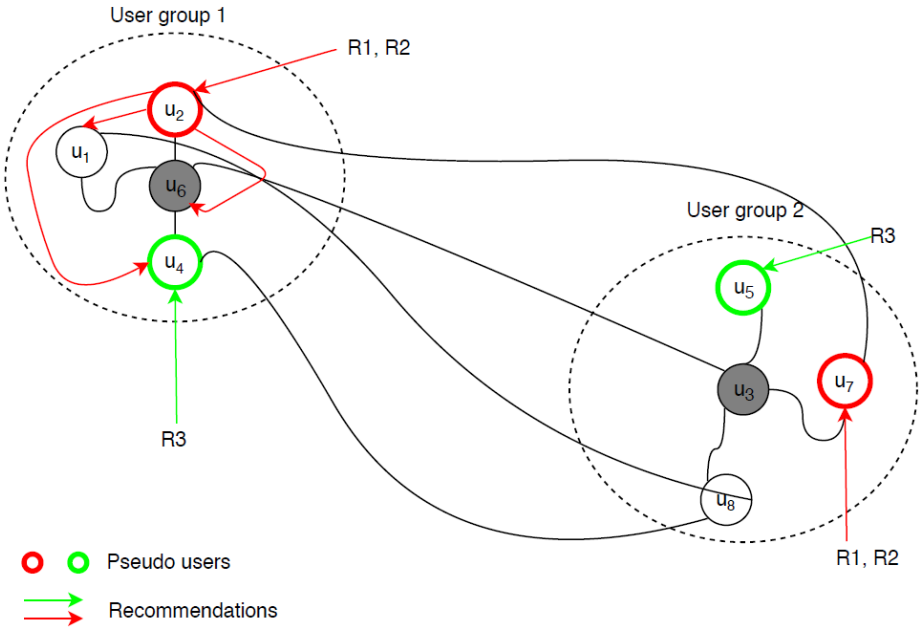


Figure 8 - Example client-side Recommendation

4.6 Privacy-preserving in the system model

In the previous sections, technical details of the working 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 the privacy preservation feature of this model. First, it is shown that the proposed user group structure can protect user privacy. Then, it is proved through theorems that user privacy will not be exposed by computations inside user groups, which are user group construction, user interest modeling, pseudo user management, and content recommendation.

4.6.1. Privacy-preserving in user groups

"Theorem 1" [14]. Let U be a set of users in an online social community ($|U| > 1$). The execution of *SecureGrouping* (Algorithm 1) on U reveals none of the interest privacy of users in U.

Proof. During the execution of *SecureGrouping*, the only step that contains communication is to choose a user group to join, in which no information exchange takes place. Therefore, the SecureGrouping protocol will not expose user privacy at all.

"Theorem 2" [14]. Let g be a user group constructed by Algorithm 1, $S_g \geq 3$. $\Pi(u_1,u_2)$ is an algorithm for user u_1 to infer the interest privacy of user u_2 in the same user group during the recommendation process (following the semi-honest behavior). Let $\Pi(u_1,u_2)$ denote the result of $u1$ executing Π on $u2$. Then, for any user, $u \in g$ and any other two users, u1u2∈g, $\Pi(u,u_1)$ and $\Pi(u,u_2)$ are perfectly indistinguishable.

Proof. Let $\{j \in I | \exists p \in pg, r_i > 0\}$ be the set of items satisfying $r_i > 0$ in g, where I is the set of items in the system. The only information that can be utilized by Π are u's input data Input u and the computation outputs Output, because all the other intermediate data are encrypted using communication protocols 1

and 2, in this system and no further information could be obtained by *u* as users in *g* are not colluding in the semi-honest model.

Then, for each item $i \in \{j \in I | \exists p \in pg, r_i > 0\}$, the information about *i* is contained in *Output(i) − Input (i)*, where *Output(i)* is the output of the computation on *i* and *Input(i)* is the input of *u* in the the computation on *i*. There are two scenarios to be discussed:

- *Output(i) − Input (i)* = 0. In this case, neither u_1 nor u_2 is interested in *i*, so that $Pr(i \in \Pi(u,u1)) = 0, Pr(i \in \Pi(u,u2)) = 0$
- *Output(i) − Input (i)* > 0. As *Output(i) − Input (i)* is the combined information of u_1, u_2 and users in $g - \{u, u1, u2\}$, so that *Input (i)* and $Input_{u2}(i)$ cannot be distinguished from *Output(i) − Input (i)* in the semi-honest model where no users are colluding, i.e., $Pr(Input_{u1}(i)) > 0) = Pr(Input_{u2}(i)) > 0)$

In both cases, we have $Pr(i \in \Pi(u,u1)) = Pr(i \in \Pi(u,u2))$

Thus, we can say that $\Pi(u, u1)$ and $\Pi(u,u2)$ have the same distribution over $\{j \in I | \exists p \in pg, r_i > 0\}$.i.e that $\Pi(u,u1)$ and $\Pi(u,u2)$ are perfectly indistinguishable.

4.6.2. Privacy protection in user interest modelling

In this model, user interests are modelled by the proposed user interest modelling algorithm. Here, we prove that the proposed user interest modelling method is privacy-preserving for users and protects users interest from adversary attacks. The first protocol is based on Elliptic-curve cryptosystem, while the second protocol is based on Shamir's secret sharing scheme.

Theorem 3. Let *U* be a set of users in an online social community ($|U| > 1$). The execution of the proposed user interest modeling method on *U* is privacy - preserving for all users in *U*.

Proof. During the user interest modeling process, communication protocol 1 and protocol 2 are the two steps required for item similarity computation. All other steps are performed by the recommender server, during which no user interest privacy could be obtained. By applying the Theorem, if we can prove that communication protocol 1 and communication protocol 2 is privacy-preserving, then we can prove that the user interest modeling method is privacy-preserving. 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 [8]:

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, by applying the Theorem 3, we can say that the proposed user interest modeling method is privacy-preserving for all users in *U*.

4.6.3. Privacy-preserving in the content recommendation

Content recommendations require association rule mining of *itemLists* received after secure distribution of users *itemLists* through communication protocol 1 and communication 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 *itemLists* from users are sent to the server via the host, and hence, the privacy of real users is protected from recommender server.

Server-side recommendation The server-side recommendation is only using the interest profiles of pseudo-users formed after the delegation of interest groups so that the privacy of real users is protected from the server. Meanwhile, it is shown that [14] user privacy can be protected inside each user group when maintaining pseudo-users. Thus, the server-side recommendation is privacy-preserving.

Client-side recommendation The client-side recommendations are all computed based on the recommendation of pseudo-users profiles, just like in [14]. Recommendations to pseudo-users contain no privacy of the real user, and users *itemLists* are stored locally on. Thus, user privacy would not be exposed to the client-side recommendation.

Bibliography

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

[2] 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.

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

[4] 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.

[5] 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.

[6] 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.

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

[8] 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.

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

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

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

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

[13] 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.

[14] 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.

[15] 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.

[16] شماره دوم 1376 "مجله جامعه شناسی ایران سال ,جامعه شناسی ایران .م," vol. 16, no. 9, p. 27, 2004.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] A. Shamir, "Shamir - 1979 - How to share a secret.pdf," pp. 612–613, 1979.