University of the Peloponnese
Department of Informatics and Telecommunications
Software and Database Systems Laboratory
https://soda.dit.uop.gr

Making recommendations in Social Networks based on textual reviews: a confidence-based approach

Technical Report TR-19002
Version 2.0
Dionisis Margaris, Costas Vassilakis, Dimitris Spiliotopoulos
margaris@di.uoa.gr, costas@uop.gr, dspiliot@uop.gr

May, 2020
Tripoli, Greece

1. Introduction

In this technical report, we present the experimental findings from applying an algorithm that (1) considers the characteristics of Social Networks (SNs) user reviews which affect the review-to-rating conversion procedure, (2) computes a confidence level for each rating, which reflects the uncertainty level for each conversion process and (3) exploit this metric both in the users' similarity computation and in the prediction formulation phases in recommender systems.

More specifically, we evaluate the performance of the proposed approach in terms of (i) SN users' satisfaction and (ii) precision, regarding the recommendations formulated based on the rating predictions generated by the proposed algorithm.

2. Social networking, semantic data management and quality of service foundations

In the following subsections we summarize the concepts and underpinnings from the areas of social networking, semantic data management, quality of service (QoS) and physical distance-based venue similarity, which are used in our work.

2.1 Influence in social networks

Within a SN, "social friends" greatly vary regarding the nature of the relationship holding among them: they may be friends or strangers, with little or nothing in between [1]. Users have friends they consider very close, and know each other in real life and acquaintances they barely know, such as singers, actors and athletes [2]. Bakshy et al. [3] suggest that a SN user responds significantly better to recommendations (e.g. advertisements) that originate from friends of the SN to which the user has a high *tie strength*. In their work, the strength of the directed tie between users i and j is linked to the amount of communication that has taken place between the users in the recent past and is computed as:

$$TS_{i,j} = \frac{C_{i,j}}{C_i} \tag{1}$$

where C_i is the total number of communications posted by user i in a certain time period (a period of 90 days is considered for computing the tie strength) in the SN, whereas $C_{i,j}$ is the total number of communications posted on the SN by user i during the same period and are directed towards user j or on posts by user j. Although the tie strength metric can be used to locate the influencers of a user, it does not consider user interests, which are important in RS. In our work, we adopt the more elaborate influence metric presented in [4], which computes the tie strength between users i and j for each distinct interest. In more detail, the influence metric $IL_{i,C}(j)$, where C is an interest category is defined as follows:

$$IL_{i,C}(j) = \begin{cases} TS_{i,j}, if C \in interests(i) \wedge C \in interests(j) \\ 0, otherwise \end{cases} \tag{2}$$

Effectively, this formula assigns a zero influence level value for interests that are not shared among the considered users, whereas for common interests, the value of the tie strength is used. For the population of each user's interest set, we use the user interest lists collected by the SN [5]. Since this list is built automatically when the user interacts with the SN, it will be comprehensive and will include all categories that the user is interested in.

Following the results presented in [4], we consider up to $N = 30$ influencers per user and only maintain influencers having an influence level ≥0.18. The set of influencers of user u in category C will be denoted as $Infl_C(u)$.

2.2 The taxonomy of venue categories

The influence level calculation scheme presented in the previous subsection relies on the allocation of venues into categories, so as to increase the granularity of the computed influence levels, aiming to increase prediction accuracy and recommendation utility. Venues categorization may be performed at different granularity levels: For example, FourSquare assigns venues to branches of a six-level taxonomy; the following list presents the correspondence between taxonomy levels and relevant information granularities, including also relevant examples:

- level 0: this level encompasses all venues.
- level 1: venue grouping at very high level. Level examples: shop & service, arts & entertainment, nightlife spot, etc.
- level 2: at this level, broad categories of venues are defined. Level examples: shopping mall, museum, bar, etc.
- level 3: at this level, broad-level categories are refined to derive more specific categories. Level examples: Accessories Store, science museum, cocktail bar, etc.
- level 4: very detailed classification of venues (available in few level 3 categories only, which are located under the "food" and "outdoors & recreation" level 1 categories). Level examples: Japanese Curry Restaurant (specialization of Japanese restaurants and, Yoga Studio (specialization of Gym / Fitness Center).
- level 5: actual venues.

Margaris et al. [6] have asserted experimentally that an optimal choice for the categorization detail level is the 3rd level of the above taxonomy (or level 2, where level 3 is unavailable), since (a) categories at this level are adequately specific to provide specialized, category-specific influence levels (b) no overfitting issues occur which would inhibit the computation of category-specific influence levels and (c) the storage space needs for recording user preferences and influence metrics at this level of granularity are limited to less than 100K per user, which can be accommodated in contemporary systems. Therefore, in this work we adopt employ a level-3 taxonomy to perform venue classification, and also store relevant user preferences at this level.

2.3 QoS parameters for venues

QoS is typically defined through attributes [7]. While a multitude of attributes that can be used for expressing a venue's QoS exist [8], in this paper will consider only the attributes cost (*c*), service (*s*) and atmosphere (*a*). This set of attributes is employed by many major travel services and websites, including Tripadvisor (http://www.tripadvisor.com) Opentable (https://www.opentable.com/); furthermore, the extension of the algorithm to include additional attributes is straightforward, hence confining the discussion to these three attributes does not lead to loss of generality.

In order to decide on which venue to visit, a user is expected to aim towards the maximization of service and atmosphere and the minimization of cost; since these goals are typically contradictory, a "golden cut" would be pursued by e.g. compromising cost optimality in favor of obtaining higher service level. Cost is usually expressed in actual currency, while a normalized indicator may also appear (e.g. from a single dollar sign for very cheap venues to five consecutive dollar signs, for very expensive ones); service and atmosphere are expressed in some scale, typically 1-5 or 1-10. In the rest of this paper we will use actual currency to represent cost and adopt the scale 1-10 for service and atmosphere. An example of the

London’s restaurants qualitative characteristics values are shown in Table 1 (values are sourced from http://www.tripadvisor.com/Restaurants-g186338-London_England.html).

Table 1. Sample QoS values within the repository

Place	cost	service	atmosphere
Restaurant Gordon Ramsay	$140	10	9
Italian Pizza Connection	$45	9	8
London Fish & Chips	$8	8	7
. . .			

2.4 Venue semantic information and similarity

The semantic information of venues can be accommodated using ontologies [6]. Under this approach, the taxonomy described in section 3.4 is enriched as follows:

- Nodes representing categories, which form a tree by virtue of the fact that these nodes form a taxonomy, are enhanced with a set of property definitions, which are applicable to all venues that are classified in the particular category (or any more specific one). A property definition lists the property name and type (e.g. integer, string, enumeration etc.). For example, the category “Nightlife spots” may specify a property “Capacity” of type “integer”, which would be applicable to all actual venues belonging in this category or any of its subcategories.
- Each leaf node may use any of the properties applicable for its category, and populate it with a specific value, compatible with the type of the property.

Having this representation available, the semantic similarity $semSim(v_i, v_j)$ between two venues v_i and v_j can be computed as follows [6]:

$$semSim(v_i, v_j) = \frac{\sum_{p \in v_i \wedge p \in v_j} sim_p(\mathrm{v_i}.p, \mathrm{v_j}.p)}{|prop(v_i) \cup prop(v_j)|} \quad (3)$$

where $\mathrm{v_i}.p$ and $\mathrm{v_j}.p$ are the values of property p for venues $\mathrm{v_i}$ and $\mathrm{v_j}$ respectively, $sim_p(\mathrm{v_i}.p, \mathrm{v_j}.p)$ is a metric of the similarity between the values of property p; finally, $prop(v_i)$ (resp. $prop(v_j)$) is the set of properties in venue $\mathrm{v_i}$ (resp. $\mathrm{v_j}$). Note that the similarity computation function is property-specific; for instance, when comparing the attribute *musicGenre* for two venues, $sim_{musicGenre}(newWave, postPunk)$ may yield *0.9* (i.e. a high value) and $sim_{musicGenre}(newWave, opera)$ may yield *0.1* (i.e. a low value). For numeric-typed properties such as ratings and costs, the $\mathrm{sim_p}$ function may be defined as:

$$sim_{num\_prop}(v1, \mathrm{v2}) = 1 - \frac{|\mathrm{v1} - \mathrm{v2}|}{\max(num\_prop) - \min(num\_prop)} \quad (4)$$

where *max(num_prop)* and *min(num_prop)* are the maximum and minimum values respectively of *numeric_prop* in the ontology extension. Equation (4) effectively corresponds a numeric value normalization formula [9]. Domain-specific similarity functions can be employed to leverage similarity calculation accuracy, e.g. Pirasteh et al. [10] introduce methods for computing metrics sim_g and sim_d, representing the similarity between movie genres and movie directors, respectively. If equation (4) cannot be used and no domain-specific similarity is available, equation (5) can be employed as a fallback similarity computation formula.

$$sim_{default}(v_1, v_2) = \begin{cases} 1, if\, v_1 = v_2 \\ 0, otherwise \end{cases} \tag{5}$$

For a more detailed discussion on venue similarity computation, the interested reader is referred to[6].

2.5 Physical distance-based venue similarity

Jones et al. [11] have demonstrated that the physical distance between venues plays an important role, since venues in close proximity are more bound to be visited by the same person, contrary to venues that are distant from each other. The computation of the physical distance-based venue similarity between the venues in [11] takes into account two factors, namely the normalized Euclidian (NED) and the normalized hierarchical distance (NHD): NHD is based on the "part-of" relation hierarchy(e.g. Wisconsin is *part-of* Dane County, which is *part-of* the State of Wisconsin, which is *part-of* the U.S.A. etc.). The two metrics are combined into a single, comprehensive metric denoted as *Total Spatial Distance* (*TSD*) using a weighted sum approach, as denoted in equation (6):

$$TSD(loc_1, loc_2) = w_e * NED(loc_1, loc_2) + w_h * NHD(loc_1, loc_2) \tag{6}$$

In equation (6), w_e and w_h represent the weights assigned to *NED* and *NHD* respectively; [11], w_e is set to 0.6 and w_h to 0.4.Based on the TSD metric (which is normalized in the range [0, 1]), we can compute physical distance-based venue similarity as

$$PhysDistSim(loc_1, loc_2) = 1 - TSD(loc_1, loc_2) \tag{7}$$

For more details, the interested reader is referred to [11]; note that in this work, a *venue thematic distance metric* is also used, however in our work thematic distance is encompassed into the semantic similarity metric.

3. Rating prediction computation

In CF, predictions for a user *U* are computed based on a set of users who have rated items similarly with *U*; this set of users is termed “near neighbors of *U*” (NNs). The similarity metric most widely used in CF-based systems is the Pearson correlation coefficient [12], where the similarity between two users *U* and *V* is expressed as:

$$sim(U,V) = \frac{\sum_{k}(\mathrm{r_{U,k}} - \overline{\mathrm{r_U}}) * (\mathrm{r_{V,k}} - \overline{\mathrm{r_V}})}{\sqrt{\sum_{k}(\mathrm{r_{U,k}} - \overline{\mathrm{r_U}})^2} * \sqrt{\sum_{k}(\mathrm{r_{V,k}} - \overline{\mathrm{r_V}})^2}} \tag{8}$$

Where *k* ranges over items that have been rated by both *U* and *V*, while $\overline{r_U}$ and $\overline{r_V}$ are the mean value or ratings entered by users *U* and V, respectively. Then, for user *U*, his NN users NNU are chosen, selecting the users having the highest similarity values with *U*. Afterwards, in order to compute a rating prediction $p_{U,i}$ for the rating of user *U* on item *i*, formula (9) is employed:

$$p_{U,i} = \overline{r_u} + \frac{\sum_{\mathrm{V} \in NN_u} sim(U,\mathrm{V}) * (r_{\mathrm{V},i} - \overline{r_{\mathrm{V}}})}{\sum_{\mathrm{V} \in NN_u} sim(U,\mathrm{V})} \tag{9}$$

The proposed algorithm modifies the prediction computation phase, by taking into account the confidence level associated with each individual rating. More specifically, formula (8) is modified as follows:

$$sim(U,V) = \frac{\sum_{k}(\mathrm{r_{U,k}} - \overline{\mathrm{r_U}}) * \mathrm{CL_{U,k}} * (\mathrm{r_{V,k}} - \overline{\mathrm{r_V}}) * \mathrm{CL_{V,k}}}{\sqrt{\sum_{k}((\mathrm{r_{U,k}} - \overline{\mathrm{r_U}}) * \mathrm{CL_{U,k}})^2} * \sqrt{\sum_{k}((\mathrm{r}_{V,\mathrm{k}} - \overline{\mathrm{r_V}}) * \mathrm{CL_{V,k}})^2}} \tag{10}$$

and formula 9 is modified as shown in equation 11:

$$p_{U,i} = \overline{r_u} + \frac{\sum_{\mathrm{V} \in NN_u} sim(U,\mathrm{V}) * (r_{\mathrm{V},i} - \overline{r_{\mathrm{V}}}) * \mathrm{CL_{V,i}}}{\sum_{\mathrm{V} \in NN_u} sim(U,\mathrm{V}) * \mathrm{CL_{V,i}}} \tag{11}$$

where *CL* is the confidence level assigned to each rating; the value of the confidence level depends on its provenance, i.e. whether it was explicitly entered or computed based on reviews; in the former case, the value of *CL* equals 1.0, while in the latter case, the value of *CL* is less than or equal to 1.0, and depends on the features of the review text. In equations 3 and 4 while $\overline{r_U}$ and $\overline{r_V}$denote the weighted average of the corresponding user’s ratings, where weighting is based on the confidence level assigned to each individual rating.
Formally, the weighted average is computed as shown in equation 12:

$$\overline{r_u} = \frac{\sum_i \mathrm{CL_{U,i}} * \mathrm{r_{U,i}}}{\sum_i \mathrm{CL_{U,i}}} \tag{12}$$

After having examined the usefulness of four different review text features as candidates to be used for improvement of prediction accuracy in the context of CF, it was determined that the polarity term density feature, i.e. the ratio of the absolute difference of positive and negative terms to the review length, was the feature most strongly associated with the textual review-to-rating conversion accuracy.

Furthermore, the optimal mapping function *CL* for the computation of the confidence level, has been found to be:

$$CL(ptd) = \begin{cases} 0.2, & if\ 0\% \le ptd < 5\% \\ 0.3, & if\ 5\% \le ptd < 10\% \\ 0.5, & if\ 10\% \le ptd < 15\% \\ 0.7, & if\ 15\% \le ptd < 17.5\% \\ 0.9, & if\ 17.5\% \le ptd < 20\% \\ 1, & if\ 20\% \le ptd \le 100\% \end{cases} \quad (13)$$

In order the aforementioned rating prediction equation (11), to be adapted in venue recommendation formulation, we modify formula (4), so as to take into account both (a) the confidence associated with each rating and (b) the influence levels between users; the modified formula is shown in equation (14):

$$p_{u,x} = \overline{r_u} + \frac{\sum_{\mathrm{v} \in NN(u)} sim(u,\mathrm{v}) * w_{u,Cat(x)}(\mathrm{v}) * CL(r_{v,i}) * (r_{u',i} - \overline{r_\mathrm{v}})}{\sum_{u' \in NN(u)} |sim(u,\mathrm{v}) * CL(r_{v,i}) * w_{u,Cat(x)}(v)|} \quad (14)$$

In this formula, again *c(r)* denotes the confidence assigned to rating *r*, whereas $w_{u,Cat(x)}(v)$ corresponds to the weight associated with the opinion of user u in relation to u' for the category that item *x* (i.e. the item for which the prediction is computed) falls in. Similarly to the approach presented in [6]), the weight is used to amplify the effect that a user's influencers have on the computation of the predictions and is defined as follows:

$$w_{u,Cat}(u') = \begin{cases} 1 + IL_{u,Cat}(v), if\ v \in Infl_{Cat}(u) \\ 1, if\ v \notin Infl_{Cat}(u) \end{cases} \quad (15)$$

adopting the formula used in the *hybrid* approach presented in [6], which is the best performing one among the options reviewed in that work, but substituting the item category-insensitive tie strength between users with the category-aware influence level discussed in subsection 2.1.

4. Venue recommendation formulation

In order to formulate a venue recommendation that considers on the one hand the opinions of the users' nearest neighbors and influencers, and on the other hand QoS and similarity aspects, two subtasks are executed in parallel, following the RS architecture presented in [6]: the first task computes a QoS-based recommendation considering only the qualitative characteristics of each venue, while the second task computes a CF-based recommendation considering the opinions of the user's nearest neighbors and influencers. Then, the two recommendations are combined to formulate the final recommendation, employing a metasearch algorithm [9], as presented in [13]. For each place category, we use a distinct set of influencers (which are stored in the user profile), aiming to leverage the accuracy of recommendation [5,6]. Once the two tasks have concluded, their results are combined into a single recommendation for the user through the use of the $WCombSUM_i$ metasearch combination formula [14], depicted in equation (15). The $WCombSUM_i$ formula computes the overall score for a venue v for a particular user u as the weighted sum of the QoS-based score $w_{QoS,C(v),u}$ and the CF-based score $w_{CF,C(v),u}$ for the particular venue.

In order to combine the QoS-based recommendation and the CF-based recommendation into a single recommendation for the user, we use the $WCombSUM_i$ formula [14]. According to this formula, the overall score for a venue v within the final recommendation for user u is

$$VenueScore_{v,u} = w_{CF,C(v),u} * score_{CF,v,u} + w_{QoS,C(v),u} * score_{QoS,v,u} \quad (16)$$

where $C(v)$ denotes the category of venue v; $w_{CF,C(v),u}$ and $w_{QoS,C(v),u}$ are weights assigned to the scores produced for venue v by the CF-based and the QoS-based algorithm, respectively.

To further promote tailoring of recommendation to individual users, the weights $w_{CF,C(v),u}$ and $w_{QoS,C(v),u}$ are both user-specific and category-specific, e.g. the weight used for the category *museums* may be different for users u_1 and u_2, while additionally the weights used for recommending a bar to user u_1 may be different than the ones employed when recommending a shopping mall to the same user.

Weight value computation is based on the assessment of how receptive a user is to the recommendations made by her influencers: the greater the receptiveness level, the higher the weight assigned to the CF-based dimension. More specifically, the values of the weights are calculated as follows:

$$\begin{aligned} w_{CF,c,u} &= \frac{|VenuesVisitedDueToInfluence_{c,u}|}{|VenuesVisited_{c,u}|} \\ w_{QoS,c,u} &= 1 - w_{CF,c,u} \end{aligned} \quad (17)$$

We can observe that the CF-based score weight $w_{CF,c,u}$ is calculated as the ratio of the number of venues within category c that user u has visited due to recommendations made to her based on her influencers' or near neighbors ratings, by the total number of places within category c that u has visited. Obviously, a ratio value close to 1 indicates that the user nearly always follows these recommendations, while a value close to 0 denotes that influencers' recommendations are disregarded by the user. In order to estimate the set $VenuesVisitedDueToInfluence_{c,u}$, we adopt the approach introduced by Margaris et al. [6], according to which a visit to a venue v by a user u is deemed to have been triggered by the user's influencers or near neighbors if (a)the system had offered to the user a recommendation for the venue prior to her visit (b) the recommendation had considered the rating entered by an influencer or near neighbor.

The computation of the CF-based score ($score_{CF,v,u}$)and the QoS-based($score_{QoS,v,u}$) referenced in equation (10) is described in the following paragraphs. The operation of the algorithm is divided in three phases: (a) *offline initialization*, where a set of metrics required for recommendation formulation is pre-computed and stored in a database to promote efficiency (b) *online operation*, where recommendations to users are formulated and (c) *repository update*, where changes in the SN status and the venue database are accommodated into the pre-computed metrics database, by recomputation of the affected metric values.

Phase 1 – Offline Initialization. The bootstrapping of the algorithm entails the following actions:

- for each venue category *c*, the minimum and maximum values for all the QoS attributes among all venues in the category are computed. The equations used for the computation of the minimum and maximum cost within a category *c* are shown in equation (18), while the calculation of the minimum and maximum service and atmosphere within a category *C* is performed in an identical fashion.

$$\begin{aligned} minCost(c) &= \min_{venue_i \in c}(cost(venue_i)) \\ maxCost(c) &= \max_{venue_i \in c}(cost(venue_i)) \end{aligned} \quad (18)$$

- for each user *u* and venue category *c*:
  - the CF-based and QoS-based weight values ($w_{CF,C(i),u}$ and $w_{QoS,c,u}$, respectively) are computed, by employing equation (17).
  - the average QoS values (cost, service and atmosphere) of the venues within category *c* that user *u* has visited in the past are computed.
  - the level of influence of her social friends for the particular category is calculated as discussed in subsection3.3, and subsequently the *top-K*ones with the highest influence levels are retained. Regarding the value of *K*, in this paper we use the value *K=6*, adopting the results of [5] which demonstrate that this setting yields optimal results.
- for each user u, the venues that she has visited are stored in her profile using a taxonomy level-3 detail.
- for each pair of venues (v_1, v_2), we compute their similarity$VenueSim(v_1, v_2)$,considering (a) the semantic similarity between (v_1, v_2) and (b) the physical distance-based similarity between the venue locations; the similarity between venues $v_1$and$v_2$is computed as:

$$VenueSim(v_1, v_2) = SemSim(v_1, v_2) * PhysDistSim\big(loc(v_1), loc(v_2)\big) \quad (19)$$

  where $SemSim(v_i, v_j)$ is the semantic similarity between venues v_i and v_j(c.f. subsection 2.4, *loc(p)* denotes the location at which *p* is located, and $PhysDistSim(loc_i, loc_j)$ corresponds to the physical distance-based similarity of locations loc_i and loc_j.

Phase 2 – Online operation: Once initialization has concluded, the online operation phase of the algorithm commences, during which recommendations are generated. Algorithm execution is triggered when a recommendation for a user *U* regarding venues in a category *C* is needed: this may be due to an express request from the user for such a recommendation, or when the SN logic considers the forwarding of such a recommendation to be appropriate.

Recommendation formulation proceeds by first computing rating predictions for all venues in category *C* that *U* has not visited insofar. For each of these venues *w*, the respective QoS-based scores $score_{QoS,w,U}$arecomputed:

$$score_{QoS,w,U} = cost\_vicin(U,w) * ser\_vicin(U,w) * \ atm\_vicin(u,W) \qquad (20)$$

where *cost_vicin(u, v)* (cost vicinity) quantifies how close the venue price is to the user's price habits within the specific category. This is computed as

$$cost\_vicin(U,w) = 1 - \frac{|cost(w) - MC(U,C)|}{maxCost(C) - minCost(C)} \qquad (21)$$

where *cost(w)* is the cost associated with venue *w* and *MC(u,C)* corresponds to the mean cost of places within category *C* that *U* visits. Correspondingly, the calculation of service vicinity and atmosphere vicinity is illustrated in equation (22):

$$ser\_vicin(U,w) \ = \begin{cases} 1 - \dfrac{|ser(w) - MS(U,C)|}{maxSer(C) - minSer(C)}, if\ ser(w) \leq MS(U,C) \\ 1, \quad if\ ser(w) > MS(U,C) \end{cases}$$
$$atm\_vicin(U,w) \ = \begin{cases} 1 - \dfrac{|atm(w) - MA(u,C)|}{maxAtm(C) - minAtm(C)}, if\ atm(w) \leq MA(U,C) \\ 1, \quad if\ atm(w) > MA(U,C) \end{cases} \qquad (22)$$

where *MS(U,C) and MA(U,C)* are the mean service and mean atmosphere respectively of places visited by *U* within *C,* and *ser(w)* and *atm(w)* are the service and atmosphere ratings. In formula (22) we can observe that when the actual value of a venue's service or atmosphere surpasses the mean value of the respective metric for the particular user and venue category, the venue is considered as totally similar to the user's profile: this stems from the fact that users always try to maximize service and atmosphere.

If the value of $score_{QoS,w,U}$ surpasses a pre-specified threshold Th_{QoS}, then the QoS parameters of venue *w* are deemed to be adequately close to the QoS levels of venues typically visited by *U*; in this respect, *w* is marked as a candidate for recommendation. In this respect, its overall score is computed by employing formula (16), and venue *v*along with its overall score is stored in the "potential recommendations" list. In this work, we use the threshold value Th_{QoS}=0.68, adopting the results of [6].

If, the QoS-based score $score_{QoS,v,u}$ is less than the Th_{QoS} threshold value (0.68), then the QoS parameters of *w* are deemed to be "not close enough" to venue visiting patterns of user *U* within category *c*, and therefore *w* is considered as not appropriate for recommendation. Taking this into account, the algorithm proceeds to find a venue *w'* which (a) satisfies the QoS requirements of user *U* and (b) is "similar" to *w*. More specifically, following steps are taken:

1. the algorithm locates within category *c* venues *w'* for which $score_{QoS,v',u}$ is greater than the threshold value Th_{QoS}=0.68. Since the QoS-based score for these venues is greater than the threshold, they can be candidates for recommendation to *U*.
2. For each such venue *w'*, the respective CF-based rating is computed as shown in equation (23):

$$score_{CF,w',u} = score_{CF,w,U} * PhysDistSim\big(loc(w), loc(w')\big) * SemSim(w,w') \qquad (23)$$

In equation (18) we can observe that the QoS-based score value for the "replacement" venue *v'* starts off with the QoS-based score value of the original venue *w* and is subsequently attenuated through the consideration of physical distance and semantic (dis)similarities between *w* and *w'*.

3. Finally, all venues *v'* identified in step 2are considered: the one having the highest score is retained and appended to the list of potential recommendations.

When all candidate venues have been examined, the *K* items having the top *K* overall scores are extracted from the list of potential recommendations and are recommended to the user; number *N* may vary, depending on the system settings.

Phase 3 – Repository update. The dynamic nature of the content of the SNs and venues information, a number of database elements and linkages need to be updated, so as to keep the database up-to-date. The cases when a database update is needed are defined below:
The updates that need to be performed are as follows:

1. Each time a new venue is stored in the venue database, the minimum and maximum QoS values for all QoS attributes within this category may need to be updated.
2. When a user check-in of a user *U* to a venue within category *C* is posted to the SN, the mean QoS attribute values of the set of places within category *C* that user *U* has visited need to be updated.
3. When a user checks in a new place, this modifies the set of places that the user has checked in; if I the check-in was triggered by a recommendation to which an influencer has contributed, then the set of places visited due to influence is also modified.
4. Each time a new venue *x* is stored in the venue database, the similarity between *x* and all other venues within the database need also to be computed.
5. Finally, when a user's categories of interest change (typically when a user visits a place belonging to a category that she has not checked-in before) or the number of communications between the user and her social acquaintances is modified, the *top-K* influencers of each user *u* within each category of interest *C* need to be computed anew.

Updates (1) and (2) are computationally inexpensive, therefore they can be performed synchronously line with the processing of the triggering event. On the other hand, steps (3)-(5) are more computationally demanding; to this end, they can be executed in batch fashion, e.g. be executed periodically.

5. Experiment results

In this section, we report on our experiments aiming to evaluate the performance of the proposed approach, in terms of (i) SN users' satisfaction and (ii) precision regarding the recommendations formulated based on the rating predictions generated by the proposed algorithm.

More specifically, we conducted an experiment, aiming at assessing the recommendation precision and participants' satisfaction regarding the recommendations they received, when the algorithms presented in sections 3 and 4, are used as a basis for rating prediction; this satisfaction level was compared to that obtained from other related algorithms. The experiment evaluated recommendation precision and user satisfaction regarding offered recommendations considering two distinct cases of SN:

1. a SN where no direct relationships among users are established and the SN is essentially directed towards the collection, organization and sharing of user-contributed content [15,16]; typical examples of such SNs are IMDB [17] and Amazon [18], and
2. a SN where direct relationships between users can be established [15,16], and these relationships are subsequently exploited by the recommendation generation algorithm; typical examples of such SNs are Facebook [19] and Twitter [20].

To assess recommendation quality, we conducted a user survey in which 50 people participated. The participants were students and staff from the community of the University of Athens, Greece, and were selected from four diverse academic departments (theater studies, physics, medicine and computer science). The users' mean age was 28 years, with a minimum of 18 years and a maximum of 51. All of the participants have been Facebook users for at least 4 years, using it for at least 6 days a week and 1 hour of use per day. Each user had registered a number of reviews or check-ins (which were complete with textual data) to Facebook; the number of reviews and check-ins ranged from 63 to 281 with a mean of 105. The minimum number of Facebook friends among the participants was 73 and the maximum was 632, with a mean of 229. The profile and review/check-in data required by the algorithms were extracted using the Facebook Graph API (https://developers.facebook.com/docs/graph-api). In order to quantify and highlight the benefits of the proposed algorithm, we have considered the following four recommendation generation algorithms:

1. A plain CF algorithm without user relationship information (plain CF, no rel): numeric ratings are produced from textual reviews based only on the Yelp rating prediction (no confidence level is used). Then, a standard CF rating prediction algorithm is applied (c.f. equation 2), and the items attaining the top-K rating predictions constitute the recommendation to the user.
2. A confidence level-enhanced algorithm without user relationship information (CL-enhanced, no rel). The proposed algorithm is used to generate rating predictions, computing and exploiting the rating confidence

level. Once rating predictions have been computed, the items attaining the top-K rating predictions constitute the recommendation to the user.

3. A plain CF algorithm exploiting user relationship information (plain CF, with rel): similarly to case (1), numeric ratings are produced from textual reviews based only on the Yelp rating prediction (no confidence level is used). However, in the rating prediction process, the tie strength [49] between users moderates the degree to which each user's opinion is taken into account in the computation of the rating prediction. As shown in [49], SN users respond considerably better to recommendations (e.g. advertisements or suggestions) that originate from friends of the SN to which the user has a high tie strength; the tie strength between users u1 and u2 is computed on the basis of the amount of communication that has taken place between the users in the recent past. Again, the top-K rating predictions constitute the recommendation to the user.
4. A confidence level-enhanced algorithm exploiting user relationship information (CL-enhanced, with rel): this is similar to case (3), however a confidence level is computed when a textual review is converted into a rating, and this confidence level is exploited in the rating prediction process.

These two cases correspond to an SN where direct relationships between users can be created, while cases (1) and (2) simulate a SN where no direct relationships between users can be established. In more detail, in the context of the experiment each participant was asked to rate 20 venue recommendations presented to her, on a scale of 1 (totally unsatisfactory) to 10 (totally satisfactory). Each of the recommendation generation algorithms (1) to (4) presented above contributed five recommendations. Recommendations were presented to the users for assessment in randomized order. If more than one algorithms recommended the same item, then the item appeared only once in the result set presented to the user, and the score given for that item was accounted to all proposing algorithms. The venues used in the recommendation formulation process were chosen to be located in Athens, Greece, to ensure a more accurate recommendation rating by the users.

Figure 1 depicts the participants' satisfaction regarding the recommendations they received, on a scale of 1 to 10, for the algorithms mentioned above. On average (last column on Figure 1 the proposed algorithm employing the textual-review-to-rating confidence level (computed using the polarity term density feature) and exploiting the SN user relationships attains an overall user satisfaction of 8.1, outperforming all other approaches. In particular, when this approach is compared with the "plain CF, with rel" algorithm, i.e. its counterpart that does not use the confidence level, which achieves an average user satisfaction of 6.9, an improvement of 17.4% is observed. Similarly, in SN contexts where no relationships among users are established, the "CL-enhanced, no rel" which uses the textual-review-to-rating confidence level outperforms the "plain CF, no rel" algorithm (which is its counterpart that does not use the confidence level) by a margin of 13.6%.

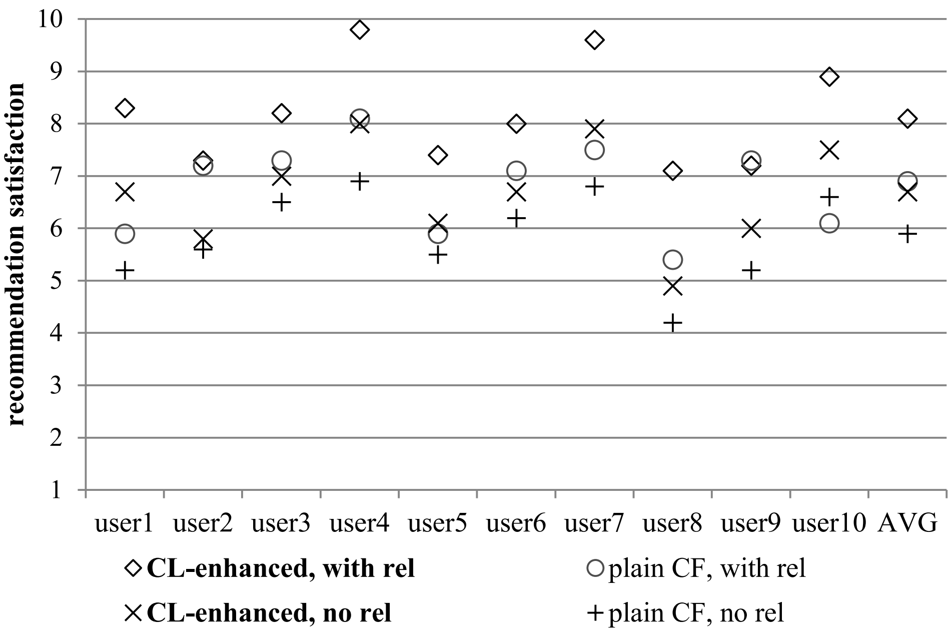

Figure 1. Users' satisfaction regarding the recommendations offered

Within Figure 1 we have also included the results regarding ten individual users; these have been chosen to demonstrate that algorithm performance is not uniform across all cases. In 92% of the cases (46 out of 50 users) the "CL-enhanced, with rel" algorithm was ranked higher than "plain CF, with rel", while in 94% of the cases (47 out of 50 users) the "CL-enhanced, no rel" algorithm was ranked higher than "plain CF, no rel". In the remaining cases, the algorithm variants not using the confidence level surpassed their counterparts that used the confidence level by a very narrow margin (up to 0.17). Further investigation on the causes that these users exhibited a different stance than the majority of users, including analysis of their profile traits, will be performed in our future work.

At the end of this report (appendix), the detailed experimental settings and results are given.

Figure 2 depicts the above-recommender-threshold precision values for the 4 settings tested in this experiment. Following the work in Felfernig et al. [21] and AlEroud and Karabatis [22], we have set the threshold to 7/10 (on our [1-10] rating scale). We can clearly see that the proposed algorithm, employing the textual-review-to-rating confidence level (computed using the PTD feature) and exploiting the SN user relationships, attains an overall precision of 81.6%, outperforming the "plain CF, with rel" algorithm by 64.5% (49.6% overall precision). Similarly, in SN contexts where no relationships among users are established, the "CL-enhanced, no rel", which uses the textual-review-to-rating confidence level, outperforms the "plain CF, no rel" algorithm (which is its counterpart that does not use the confidence level) by a margin of 23.7% (48% versus 38.8%).

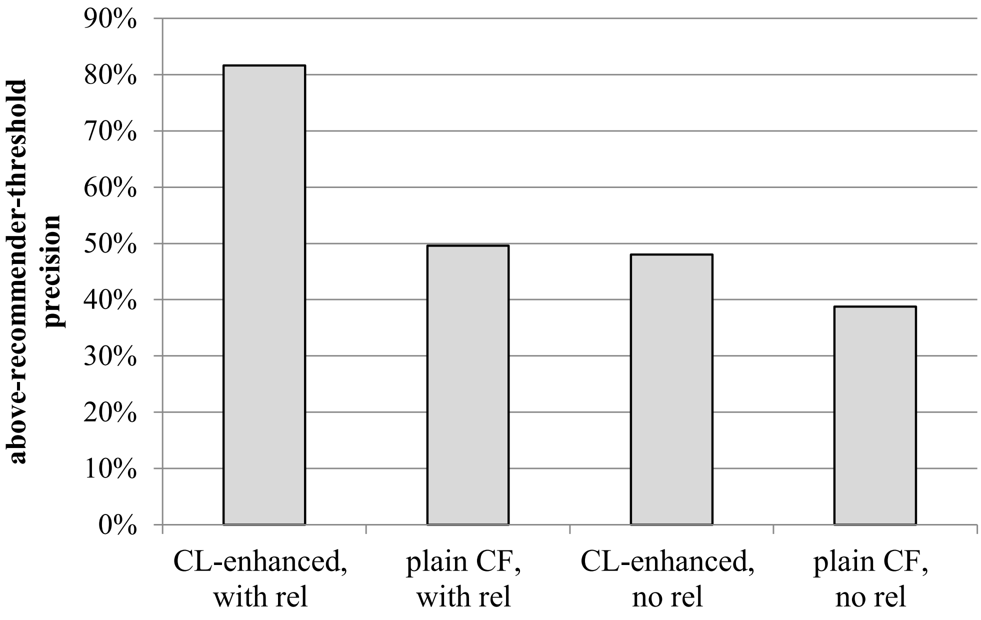

Figure 2. SN users' recommendation precision

6. Conclusions

In this report we have evaluated the performance of an algorithm that (1) considers the characteristics of SN user reviews which affect the review-to-rating conversion procedure, (2) computes a confidence level for each rating, which reflects the uncertainty level for each conversion process and (3) exploit this metric both in the users' similarity computation and in the prediction formulation phases in recommender systems, in terms of (i) SN users' satisfaction regarding venues recommendations formulated based on the rating predictions generated by it and (ii) above-recommender-threshold precision for the generated recommendations.

The results indicate that the above algorithm raises user satisfaction by a margin ranging from 13.6% to 17.4%, when compared to algorithms that do not take into account the uncertainty inherent in textual review-to-rating conversions. The respective improvement, as far as the recommendation precision is concerned, has been computed from 23.7% to 49.6%.

The introduction of the confidence level has been shown to deliver performance benefits both in SN where relationships between users can be established and in SN where such relationships are not present.

7. References

[1] E. Gilbert, and K. Karahalios, Predicting tie strength with social media, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2009), 211-220.

[2] U. Shardanand and P. Maes, Social Information Filtering: Algorithms for Automating "Word of Mouth", Proceedings of the 1995 SIGCHI Conference on Human Factors in Computing Systems (1995), 210-217.

[3] E. Bakshy, D. Eckles, R. Yan, I. Rosenn, Social Influence in Social Advertising: Evidence from Field Experiments, Proceedings of the 13th ACM Conference on Electronic Commerce (2012), 146-161.

[4] D. Margaris, C. Vassilakis and P. Georgiadis, Query personalization using social network information and collaborative filtering techniques, Future Generation Computer Systems, 78, 1 (2018), 440–450.

[5] D. Margaris, C. Vassilakis and P. Georgiadis, Recommendation Information Diffusion in Social Networks Considering User Influence and Semantics, Social Network Analysis and Mining, 6, 108 (2016), 1-22.

[6] D. Margaris, C. Vassilakis and P. Georgiadis, Knowledge-Based Leisure Time Recommendations in Social Networks, Current Trends on Knowledge-Based Systems: Theory and Applications (2016), 24–48.

[7] ITU, Recommendation E.800 Quality of service and dependability vocabulary, Blue Book, Fascicle II.3 (1998).

[8] T. Mersha and V. Adlakha, Attributes of Service Quality: The Consumers′ Perspective, International Journal of Service Industry Management, 3, 3 (1992), 34–45.

[9] J. Aslam, and M. Montague, Models for metasearch, Proceedings of the 24th annual international conference on Research and development in information retrieval (2001), 276-284.

[10] P. Pirasteh, J. Jung and D. Hwang, Item-Based Collaborative Filtering with Attribute Correlation: A Case Study on Movie Recommendation,Proceedings of the 6th Asian Conference, ACIIDS (2014), 245-252.

[11] C.B. Jones, H. Alani and D. Tudhope, Geographical information retrieval with ontologies of place. Proceedings of the 2001 Conference on Spatial Information Theory (2001), 322–335.

[12] J. Herlocker, J. Konstan, L. Terveen and J. Riedl, Evaluating Collaborative Filtering Recommender Systems, ACM Transactions on Information Systems, 22, 1 (2004), 5–53.

[13] D.Margaris, C. Vassilakis and P. Georgiadis, An integrated framework for adapting WS-BPEL scenario execution using QoS and collaborative filtering techniques, Science of Computer Programming, 98 (2015), 707–734.

[14] D. He and D. Wu, Toward a Robust Data Fusion for Document Retrieval,Proceedings of the 4th IEEE International Conference on Natural Language Processing and Knowledge Engineering (2008), 1-4.

[15] J.A. Obar and S. Wildman, Social media definition and the governance challenge: An introduction to the special issue, Telecommunications Policy, 39, 9 (2015), 745–750.

[16] M. Boyd and N. Ellison, Social Network Sites: Definition, History, and Scholarship, Journal of Computer-Mediated Communication, 13, 1 (2007), 210–30.

[17] IMDB home page, Retrieved on April 30, 2019. https://www.imdb.com/.

[18] Amazon home page, Retrieved on April 30, 2019. https://www.amazon.com.

[19] Facebook home page, Retrieved on April 30, 2019. https://www.facebook.com.

[20] Twitter home page, Retrieved on April 30, 2019. https://twitter.com/.

[21] A. Felfernig, L. Boratto, M. Stettinger and M. Tkalčič, Evaluating group recommender systems. Group Recommender Systems (2018), 59-71.

[22] A. AlEroud and G. Karabatis, A System for Cyber Attack Detection Using Contextual Semantics. Proceedings of the 7th International Conference on Knowledge Management in Organizations: Service and Cloud Computing. Advances in Intelligent Systems and Computing, vol 172.

Appendix

In this appendix, the detailed experimental settings and results of the social network user satisfaction experiment are given.

For each user, his initials are given, including his/her gender, the University Department that the user belongs to (Physics, Medicine, Theater Studies and Computer Science), along with the average value of the 5 recommendations formulated and presented to him/her, for the 4 algorithms tested.
Algorithm1 represents the *CL-enhanced, with rel* setting.
Algorithm2 represents the *plain CF, with rel* setting.
Algorithm3 represents the *CL-enhanced, no rel* setting.
Algorithm4 represents the *plain CF, no rel* setting.

USER 1: NAME K.K, GENDER M, department P
Algorithm1: 8.3
Algorithm2: 5.9
Algorithm3: 6.7
Algorithm4: 5.2

USER 2: NAME T.X, GENDER F, department M
Algorithm1: 7.3
Algorithm2: 7.2
Algorithm3: 5.8
Algorithm4: 5.6

USER 3: NAME K.V, GENDER F, department P
Algorithm1: 8.2
Algorithm2: 7.3
Algorithm3: 7.0
Algorithm4: 6.5

USER 4: NAME M.S, GENDER M, department C
Algorithm1: 9.8
Algorithm2: 8.1
Algorithm3: 8.0
Algorithm4: 6.9

USER 5: NAME C.A, GENDER M, department C
Algorithm1: 7.4
Algorithm2: 5.9
Algorithm3: 6.1
Algorithm4: 5.5

USER 6: NAME C.R, GENDER M, department P
Algorithm1: 8.0
Algorithm2: 7.1
Algorithm3: 6.7
Algorithm4: 6.2

USER 7: NAME X.Z, GENDER F, department T
Algorithm1: 9.6
Algorithm2: 7.5
Algorithm3: 7.9
Algorithm4: 6.8

USER 8: NAME E.G, GENDER F, department T
Algorithm1: 7.1
Algorithm2: 5.4
Algorithm3: 4.9
Algorithm4: 4.2

USER 9: NAME M.A, GENDER F, department T
Algorithm1: 7.2
Algorithm2: 7.3
Algorithm3: 6.0
Algorithm4: 5.2

USER 10: NAME A.T, GENDER M, department T
Algorithm1: 8.9
Algorithm2: 6.1
Algorithm3: 7.5
Algorithm4: 6.6

USER 11: NAME P.D, GENDER F, department C
Algorithm1: 8.2
Algorithm2: 6.2
Algorithm3: 6.8
Algorithm4: 6.1

USER 12: NAME A.C, GENDER F, department M
Algorithm1: 7.2
Algorithm2: 7.5
Algorithm3: 6.8
Algorithm4: 5.1

USER 13: NAME A.S, GENDER M, department P
Algorithm1: 8.3
Algorithm2: 6.3
Algorithm3: 6.7
Algorithm4: 5.8

USER 14: NAME T.B, GENDER F, department T
Algorithm1: 8.1
Algorithm2: 7.0
Algorithm3: 6.3
Algorithm4: 5.8

USER 15: NAME K.R, GENDER F, department P
Algorithm1: 7.8

Algorithm2: 6.6
Algorithm3: 7.0
Algorithm4: 6.1

USER 16: NAME S.Y, GENDER M, department T
Algorithm1: 7.7
Algorithm2: 6.7
Algorithm3: 7.1
Algorithm4: 6.6

USER 17: NAME E.A, GENDER M, department P
Algorithm1: 8.5
Algorithm2: 7.6
Algorithm3: 6.0
Algorithm4: 5.7

USER 18: NAME J.G, GENDER M, department C
Algorithm1: 8.1
Algorithm2: 6.6
Algorithm3: 6.7
Algorithm4: 5.5

USER 19: NAME H.T, GENDER F, department C
Algorithm1: 7.6
Algorithm2: 7.8
Algorithm3: 6.2
Algorithm4: 6.4

USER 20: NAME N.K, GENDER F, department M
Algorithm1: 8.8
Algorithm2: 6.6
Algorithm3: 7.6
Algorithm4: 6.3

USER 21: NAME T.E, GENDER M, department P
Algorithm1: 7.8
Algorithm2: 6.8
Algorithm3: 6.2
Algorithm4: 6.2

USER 22: NAME M.G, GENDER F, department P
Algorithm1: 8.0
Algorithm2: 7.3
Algorithm3: 6.7
Algorithm4: 5.7

USER 23: NAME M.J, GENDER M, department P
Algorithm1: 8.4
Algorithm2: 7.0
Algorithm3: 7.4

Algorithm4: 6.4

USER 24: NAME O.A, GENDER F, department P
Algorithm1: 8.1
Algorithm2: 6.5
Algorithm3: 7.1
Algorithm4: 5.8

USER 25: NAME E.V, GENDER F, department M
Algorithm1: 8.2
Algorithm2: 6.8
Algorithm3: 6.9
Algorithm4: 6.3

USER 26: NAME P.P, GENDER M, department P
Algorithm1: 8.0
Algorithm2: 6.4
Algorithm3: 6.3
Algorithm4: 5.1

USER 27: NAME A.O, GENDER M, department P
Algorithm1: 8.4
Algorithm2: 7.8
Algorithm3: 6.7
Algorithm4: 6.0

USER 28: NAME N.X, GENDER F, department P
Algorithm1: 7.1
Algorithm2: 7.1
Algorithm3: 6.8
Algorithm4: 6.7

USER 29: NAME L.A, GENDER M, department P
Algorithm1: 8.1
Algorithm2: 6.3
Algorithm3: 6.3
Algorithm4: 6.3

USER 30: NAME S.R, GENDER M, department T
Algorithm1: 7.5
Algorithm2: 7.1
Algorithm3: 6.3
Algorithm4: 5.0

USER 31: NAME S.Z, GENDER F, department P
Algorithm1: 7.4
Algorithm2: 7.6
Algorithm3: 6.9
Algorithm4: 5.9

USER 32: NAME T.S, GENDER F, department M
Algorithm1: 8.5
Algorithm2: 7.3
Algorithm3: 6.5
Algorithm4: 5.2

USER 33: NAME K.A, GENDER F, department M
Algorithm1: 8.3
Algorithm2: 6.8
Algorithm3: 7.2
Algorithm4: 6.0

USER 34: NAME K.G, GENDER F, department M
Algorithm1: 7.9
Algorithm2: 6.9
Algorithm3: 6.6
Algorithm4: 6.9

USER 35: NAME M.F, GENDER F, department T
Algorithm1: 8.6
Algorithm2: 7.2
Algorithm3: 8.0
Algorithm4: 5.1

USER 36: NAME T.T, GENDER F, department P
Algorithm1: 8.5
Algorithm2: 6.6
Algorithm3: 6.2
Algorithm4: 6.2

USER 37: NAME M.T, GENDER F, department T
Algorithm1: 8.4
Algorithm2: 6.4
Algorithm3: 7.6
Algorithm4: 6.4

USER 38: NAME V.A, GENDER F, department P
Algorithm1: 8.7
Algorithm2: 6.6
Algorithm3: 5.7
Algorithm4: 5.7

USER 39: NAME I.E, GENDER M, department T
Algorithm1: 8.4
Algorithm2: 7.0
Algorithm3: 7.7
Algorithm4: 5.2

USER 40: NAME N.Z, GENDER M, department C
Algorithm1: 8.0

Algorithm2: 6.5
Algorithm3: 6.7
Algorithm4: 6.7

USER 41: NAME T.R, GENDER M, department P
Algorithm1: 7.6
Algorithm2: 7.1
Algorithm3: 6.6
Algorithm4: 5.5

USER 42: NAME M.I, GENDER M, department T
Algorithm1: 8.1
Algorithm2: 6.6
Algorithm3: 6.5
Algorithm4: 6.3

USER 43: NAME A.G, GENDER M, department M
Algorithm1: 8.0
Algorithm2: 6.8
Algorithm3: 6.9
Algorithm4: 6.1

USER 44: NAME V.E, GENDER F, department C
Algorithm1: 8.1
Algorithm2: 7.1
Algorithm3: 7.6
Algorithm4: 4.2

USER 45: NAME A.P, GENDER M, department P
Algorithm1: 9.0
Algorithm2: 7.0
Algorithm3: 6.0
Algorithm4: 6.0

USER 46: NAME K.X, GENDER F, department C
Algorithm1: 8.3
Algorithm2: 6.1
Algorithm3: 6.9
Algorithm4: 6.1

USER 47: NAME A.T, GENDER F, department P
Algorithm1: 8.9
Algorithm2: 6.8
Algorithm3: 6.2
Algorithm4: 5.3

USER 48: NAME I.E, GENDER F, department P
Algorithm1: 8.2
Algorithm2: 6.2
Algorithm3: 6.4

Algorithm4: 6.4

USER 49: NAME E.A, GENDER F, department T
Algorithm1: 7.5
Algorithm2: 7.6
Algorithm3: 6.7
Algorithm4: 6.9

USER 50: NAME Z.R, GENDER M, department C
Algorithm1: 8.2
Algorithm2: 6.6
Algorithm3: 6.2
Algorithm4: 5.0

AVG of all users:
Algorithm1: 8.1
Algorithm2: 6.9
Algorithm3: 6.7
Algorithm4: 5.9