CS711A: Introduction to Game Theory and Mechanism Design **Fall 2020**

Project Report: Course recommender for IITK students

Group members: Bhavy Singh, Chinmaya Singal, Nallapati Sathvik, Saqib Mohammed and Varchasv Shri

Abstract

We aim to design a recommender system for a university student based on his/her course record using collaborative filtering and pareto dominance, which outputs a list of recommended courses for the user and host this system on a web-page to increase it's accessibility. A student has to provide the courses s/he has liked or disliked till now. Our system uses this as an input to build a profile and suggests courses based on choices and ratings of peers at university. Since one tends to give higher weightage to information provided by acquainted friends or seniors from their department, we introduce the concept of trust in our recommender where the trust statements denote the department of their interest. Pareto dominance concept is used to calculate the effectiveness factor of each trusted user of the active user. Those who are non-dominated have a higher effect on recommendation than those of the dominated users. To evaluate the usability of this method, we analyze real-world data obtained from student registration in IIT Kanpur.

1 Introduction to the problem

Every semester, all students of IIT Kanpur have to enroll for a minimum of 35 credits, or in terms of courseload, approximately 4 courses. If however, a student fails to meet this minimum cutoff, s/he cannot register for the next semester. While there are a myriad of courses to choose from as far as electives are concerned, the issue is that the students can be confused by the variety, and usually just end up pursuing electives that either have very lenient grading as far as their seniors are concerned, or are open for mass registration.
These however, might not be courses that the students themselves are interested in, which was our motivation in making our course recommender. Instead of relying only the past experiences of a select few students that every student ends up talking to, we aim to draw upon the generalized experience of every student enrolled in the institute. Thereby, we can help them make better and more informed choices, enriching their university stay, and thus improving the quality of their college life and maybe influence their professional life by virtue of discovering and learning new skills.

1.1 Related work

Collaborative filtering is an algorithm wherein an active user is given recommendations based on the likes and dislikes of other users, weighted by the similarity between the preferences of other users. This simple intuition is effective in generating recommendations and is widely used in a variety of fields [RV97]. Chavarriaga et al.[C14] used collaborative filtering to obtain recommendations for a student given social knowledge, and established that social knowledge and a student's qualifications are a source of valuable recommendations. Bydžovská [B15] provided experimental evidence that the collaborative filtering approach is suitable for predicting student performance and [B16] showed that one can utilize similarities between students to get great results for optional course recommendations. Al-Badarenah and Alsakran [AJ16] provided a collaborative recommendation system based on similarities and dissimilarities among user preferences. Ogunmakinde et al. [O17] showed a strong positive correlation between students' satisfaction with their tutors and grades

awarded to them and also identified the grading system adopted as the main factor affecting students' satisfaction with grades.
Massa and Avesani [M09] proposed an enhanced Recommender Systems by use of trust information and named them as Trust-aware Recommender Systems. Empirical evidences were provided to establish that trust was very effective in alleviating weaknesses of recommender systems. Gupta and Nagpal [GN15] established that incorporation of trust in traditional collaborative filtering, achieved better performance by minimizing overall prediction error. Marsh [S94] introduced trust as a computational concept. He also introduced the concept of distrust as the negative trust. Gupta and Nagpal [SS15] showed that trust establishment can be either explicit or implicit. Explicit networks are built with explicit trust statements, which are directly provided by a user for another user. Whereas implicit trust scores are inferred from user 's behavior.
Guha et al. developed a framework of trust propagation schemes. The system assumes that the trust values of other users are specifically specified by users. In their study, they have also presented the notion of distrust and the propagation of distrust. Golbeck et al. applied the concept of the trust to the social network. They explained how trust can be measured and how it can be used in applications. They suggested algorithms to infer the relationships of trust between people who are not directly linked to the network.
Azadjalal et al. [MPAM17] used Pareto dominance and confidence concepts to identify the most prominent users of which opinions are employed in the recommendation process which showed significant improvement in accuracy measures of recommender systems. Ortega et al. [FJJA13] used Pareto dominance to perform a pre-filtering process and eliminated less representative users while retaining the most promising ones.

We used collaborative filtering to find user-user similarity based on course satisfaction which can depend on course grades and users' liking and disliking of a course. We used explicit trust statement from users which demand the list of departments the user in most interested in (Which department students the user will trust the most). We also used pareto dominance to give higher weightage to recommendation of non-dominated students. We have also built upon the existing methods to calculate trust and dominance when data such as ratings of users for each course are not available to make a recommendation.

1.2 Brief overview of the report

In Section 2 we give a brief overview of the input, output and some notation that would be used in further sections. In Section 3 we explain the main methodology used for calculating the trust values for students and then using that to calculate a score for each course. In Section 4 we provide some examples and discuss the accuracy of the results obtained from our method. Finally, we conclude this report in Section 5 by giving a brief summary of our observations and accuracy of the methodology used.

2 Model of the input, output and notation used

2.1 Input and Output

The user inputs the set of courses they liked and courses they disliked courses along with the preferred departments they are interested in.

- *uliked* - The set of courses they liked.
- *udisliked* - The set of courses they didn't like.
- $pref\_dept$ - They set of departments they are interested in. This set can contain upto 3 elements.

Using the input provided by the user, we calculate the trust, confidence and similarity value for each of the students in the institute (Y15 batch onward) with respect to our user and using those calculate a rating for each of the courses. The user gets a set of recommended courses sorted in the descending order of their rating R (a score generated by our algorithm).

2.2 Some Notations

We will calculate the similarity, confidence and trust value for each of the students w.r.t. our user which will help us in generating a score of each available course that is to be recommended. Some of the notations which will be used in further sections is as follows:

- $s_u(v)$ - Similarity value of student v w.r.t. user u.
- $c_u(v)$ - Confidence value of student v w.r.t. user u.
- $trust_u(v)$ - Trust valued of student v w.r.t. user u.
- S - The set of all the students excluding the user.
- C_v - The set of courses done by student v.
- $L_{(u,v)}$ - Set of courses done by student v that are liked by user u, i.e., $L_{(u,v)} = C_v \cap uliked$
- $D_{(u,v)}$ - Set of courses done by student v that are disliked by user u, i.e., $D_{(u,v)} = C_v \cap udisliked$
- $I_{(u,v)}$ - Set of courses done by student v that are either liked or disliked by user u, i.e., $L_{(u,v)} \cup D_{(u,v)}$
- $r_{u,i}$ - Rating given by user u to course i. It is 1 if $i \in uliked$ and -1 if $i \in udisliked$

3 Main methodology used

3.1 Similarity calculation

The similarity value of student $v \in S$ w.r.t. user u is calculated by the Pearson correlation coefficient given by the formula:

$$s_u(v) = \frac{\sum_{i \in I_{(u,v)}} (r_{u,i} - \bar{r}_u) * (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{(u,v)}} (r_{u,i} - \bar{r}_u)^2 * \sum_{i \in I_{(u,v)}} (r_{v,i} - \bar{r}_v)^2}} \tag{1}$$

Here $\bar{r}_u$ and $\bar{r}_v$ are the average ratings given by users u and v. In our implementation, since we don't know the exact rating provided by the students for each of the courses, we have estimated them by using the average grade for the offering of the course and normalise it to get a value between 0 and 1 to get $r_{v,i}$ (Because the user input rating is either -1 or 1). Hence, we will use a modified version of the Pearson correlation coefficient where we will put the average rating of the user as $\bar{r}_u = \bar{r}_v = 0$.

3.2 Confidence calculation

The confidence value of student v w.r.t. user u indicates the common interests of student v w.r.t. user u. We take the number of common liked courses, subtract the number of common disliked courses after normalization and then normalise the result with the number of liked courses of user u. The formula used for calculating $c_u(v)$ for a given student $v \in S$ is as follows:

$$c(v|u) = \frac{|L_{(u,v)}| - \frac{|D_{(u,v)}|}{|udisliked|}}{|uliked|}$$

3.3 User Dominance

A student v_1 is said to dominate a student v_2 with respect to active user u if:

$$|r_{u,i} - r_{v_1,i}| \leq |r_{u,i} - r_{v_2,i}| \ \forall i \in uliked \cup udisliked \qquad \text{and}$$

$$\exists i \in uliked \cup udisliked \text{ s.t. } |r_{u,i} - r_{v_1,i}| < |r_{u,i} - r_{v_2,i}|$$

That is, the ratings of student v_1 are at least as close to user u as that of student v_2 for all the courses and for at least one of the courses, the rating of student v_1 is strictly closer to user u as compared to student v_2.

A student v is said to be non-dominated if $\forall w \in S$, student w doesn't dominate v.
In our implementation, since we don't know the exact rating provided by the students for each of the courses, we have assigned a rating of $r_{v,i} = 1$ if $i \in C_v$

$$r_{v,i} = \begin{cases} 1 & \text{if } i \in C_v \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

3.4 Trust calculation

The trust value between user u and student v is calculated differently for dominated and non-dominated user v, using the formula:

$$trust_u(v) = \begin{cases} \mathbf{k} \times \mathbf{t_{u,v}} \times \mathbf{w_{u,v}} & \text{if v} \in NonDominated_u \\ \mathbf{t_{u,v}} \times \mathbf{w_{u,v}} & \text{if v} \in Dominated_u \end{cases} \tag{3}$$

- k is a hyperparameter which gives a higher trust value to non-dominated users. In our implementation, it is set at 3.
- $t_{u,v}$ is a base trust value assigned to each student v using the preferred department list of user u as follows:
  1. If v belongs to the most preferred department of user u, then $t_{u,v} = 1$
  2. If v belongs to the second most preferred department of user u, then $t_{u,v} = 0.8$
  3. If v belongs to the third most preferred department of user u, then $t_{u,v} = 0.6$
  4. Otherwise, it has a default value $t_{u,v} = 0.2$
- $w_{u,v}$ is a trust weight calculated using the similarity and confidence values of student v. Here, k' is a hyperparameter which is set at 0.2 in our implementation.

$$w_{u,v} = \begin{cases} \dfrac{2 \times s_u(v) \times c_u(v)}{s_u(v) + c_u(v)} & \text{if } s_u(v) \neq 0 \text{ and } c_u(v) \neq 0 \\ k' \times c_u(v) & \text{if } \; s_u(v) = 0 \text{ and } c_u(v) \neq 0 \\ 0 & \text{if } \; s_u(v) = 0 \text{ and } c_u(v) = 0 \end{cases} \tag{4}$$

3.5 Score calculation

Once we have calculated the trust value for each student v, we calculate the score/rating for each course c. To do so, first we need to calculate the average grade obtained in that course, μ(c) where we use the grade values as is used for CPI calculation.

Let A_c be the set of students who have done the course c, i.e., $v \in A_c$ iff $c \in C_v$. The score of a course c with respect to user u is given by:

$$Score(u, c) = \frac{\mu(c) \times \sum_{v_i \in A_c} trust_u(v_i)}{\sum_{v_i \in A_c} N(v_i)} + b(c) \tag{5}$$

where $N(v_i)$ denotes whether the trust value of student v_i is greater than 0 or not. Hence

$$N(v_i) = \begin{cases} 1 & \text{if } trust_u(v_i) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$b(c)$ represents a popularity bias, in other words it is higher for courses pursued by many students. Hence, it is proportional to the number of students (v_i) who have done course c.

$$b(c) = \frac{|A_c|}{H} \tag{7}$$

where H is a hyper-parameter. In our implementation, it is set at 400.
Finally, we sort the list of courses in decreasing order of the score assigned to them and suggest the top 50 courses from that list.

4 Experiments/Simulations

To test the accuracy of the recommendations, we used two methods which are as follows:

1. **Peer Feedback:** We asked some of our peers for their list of liked courses and the departments they are interested in, ran the algorithm on input provided by them and asked for their feedback about how the recommendations aligned with their interest. The feedback we received was highly positive and they even stated that the recommendations included some of the courses they are planning to do in future. One such example is as follows:
   - Input: $uliked$ =["CS637A", "CS771A", "EE609A", "EE301A", "EE627A"], $pref\_dept$ =["EE", "CSE"]
   - The output included courses related to the provided input such as CS698X, EE321A, EE698V, EE698C, EE629A, EE602A, CS698O, etc. It also included the courses they had provided in input but not done yet since we only filtered out the courses already done by the user.

2. **Cross-Validation with past data:** We used the data from past batches to check whether the recommendations aligned with what the students pursued later. To do so, we provided the input of liked courses from the courses a student had done in their first two years without providing their roll number and checked whether the list of recommended courses matched with what they pursued in their third and fourth years. The results obtained were promising and the algorithm was indeed able to recommend the courses they actually ended up doing with high accuracy. One such example is as follows:
   - Input: $uliked$ =["CS201A","CS202A","ESC201A","MSO202A","ESO204A"], $pref\_dept$ = ["CSE","MTH","PHY"]
   - Actual courses done by roll no 160538 in third and fourth year
     DC- CS252A, CS330A, CS340A, CS345A, CS335A
     UGP- CS395A, CS396A
     DE - CS648A, CS771A, CS315A, CS731, CS425A, CS445A
     OE - MTH517A, PHY305A, ECO764A, MTH628A, PHY226B
   - Our recommendation:
     All of the DC courses, UGPs like CS498, CS396A, 5 out of 6 DEs and 3 out of 5 OEs were present in the list of our recommended courses.

The screenshots of the above experiments are attached in the Appendices. The full implementation can be found here.

5 Summary and Discussions

In section 1, the main motivation behind the project, other related work, and some existing problems were discussed. As such, in sections 2 and 3, the implementation was discussed, along with how the ratings are predicted to inform the user of related courses. Finally, in section 4, methods to test the accuracy of the recommendations provided were discussed, and as shown, the system has reasonably high accuracy given the constraints.

A good future direction for this work would be to conduct survey to obtain and store student data so as to have data about the ratings of each course provided by the students. This would help make the predictions better and even more accurate.

References

[AJ16] Amer Al-Badarenah and Jamal Alsakran. "An Automated Recommender System for Course Selection". In: *International Journal of Advanced Computer Science and Applications* 7.3 (2016). DOI: 10.14569/IJACSA.2016.070323. URL: http://dx.doi.org/10.14569/IJACSA.2016.070323.

[B15] Hana Bydžovská. "Are Collaborative Filtering Methods Suitable for Student Performance Prediction?" In: *Progress in Artificial Intelligence*. Ed. by Francisco Pereira et al. Cham: Springer International Publishing, 2015, pp. 425–430. ISBN: 978-3-319-23485-4.

[B16] Hana Bydzovská. "Course Enrollment Recommender System". In: *EDM*. 2016.

[C14] Oscar Chavarriaga, Beatriz Florian-Gaviria, and Oswaldo Solarte. "A Recommender System for Students Based on Social Knowledge and Assessment Data of Competences". In: *Open Learning and Teaching in Educational Communities*. Ed. by Christoph Rensing et al. Cham: Springer International Publishing, 2014, pp. 56–69. ISBN: 978-3-319-11200-8.

[FJJA13] Fernando Ortega et al. "Improving collaborative filtering-based recommender systems results using Pareto dominance". In: *Information Sciences* 239 (2013), pp. 50–61. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2013.03.011. URL: http://www.sciencedirect.com/science/article/pii/S0020025513002004.

[GN15] S. Gupta and S. Nagpal. "An empirical analysis of implicit trust metrics in recommender systems". In: *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. 2015, pp. 636–639. DOI: 10.1109/ICACCI.2015.7275681.

[M09] Paolo Massa and Paolo Avesani. "Trust Metrics in Recommender Systems". In: *Computing with Social Trust*. Ed. by Jennifer Golbeck. London: Springer London, 2009, pp. 259–285. ISBN: 978-1-84800-356-9. DOI: 10.1007/978-1-84800-356-9_10. URL: https://doi.org/10.1007/978-1-84800-356-9_10.

[MPAM17] Mohammad Mahdi Azadjalal et al. "A trust-aware recommendation method based on Pareto dominance and confidence concepts". In: *Knowledge-Based Systems* 116 (2017), pp. 130–143. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2016.10.025. URL: http://www.sciencedirect.com/science/article/pii/S0950705116304208.

[O17] Olabode Ogunmakinde et al. "Factors Affecting Construction Students' Satisfaction with Grades in Design Courses". In: July 2017. DOI: 10.29007/xhmv.

[RV97] Paul Resnick and Hal R. Varian. "Recommender Systems". In: *Commun. ACM* 40.3 (Mar. 1997), pp. 56–58. ISSN: 0001-0782. DOI: 10.1145/245108.245121. URL: https://doi.org/10.1145/245108.245121.

[S94] Stephen Paul Marsh. *Formalising trust as a computational concept*. Tech. rep. 1994.

[SS15] S. Gupta and Sushama Nagpal. "Trust Aware Recommender Systems : A Survey on Implicit Trust Generation Techniques". In: 2015.

Appendices

A Peer Feedback Example

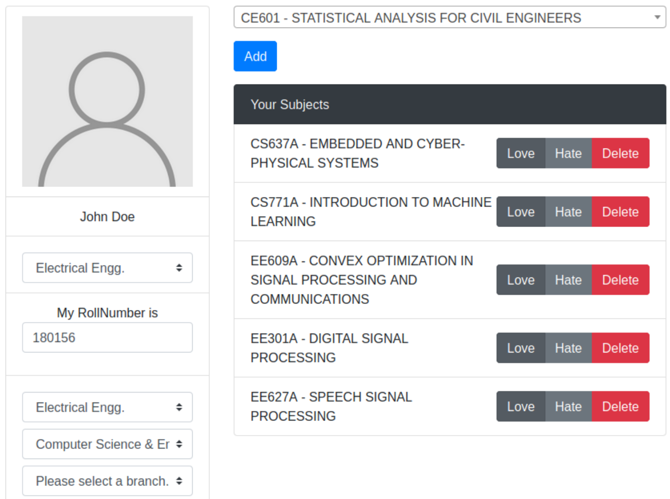

Figure 1: Input provided by user

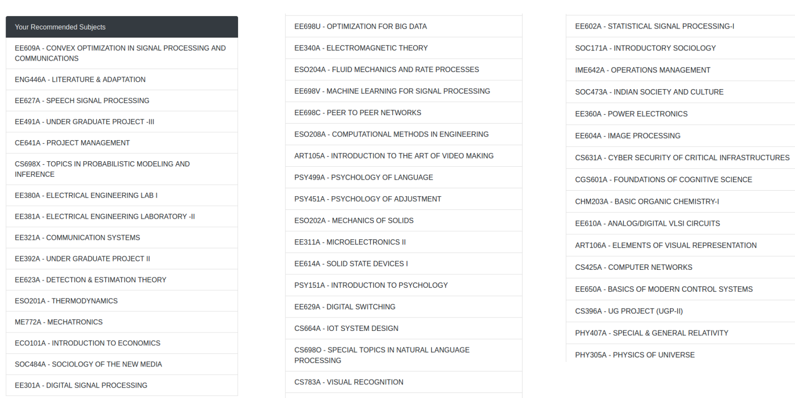

Figure 2: Recommendation output

B Cross Validation Example

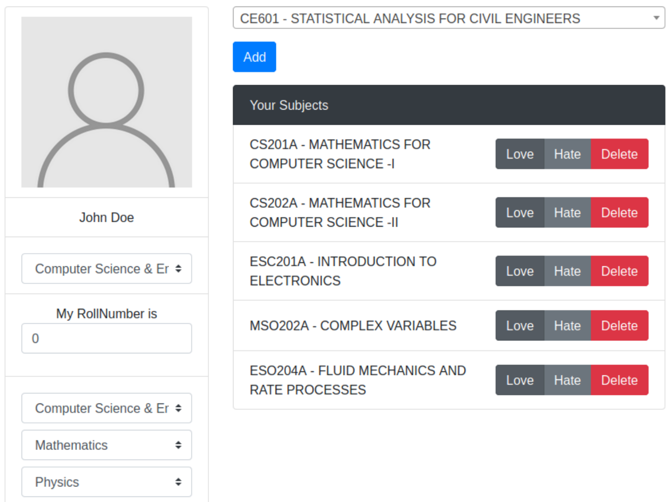

Figure 3: Input provided by user

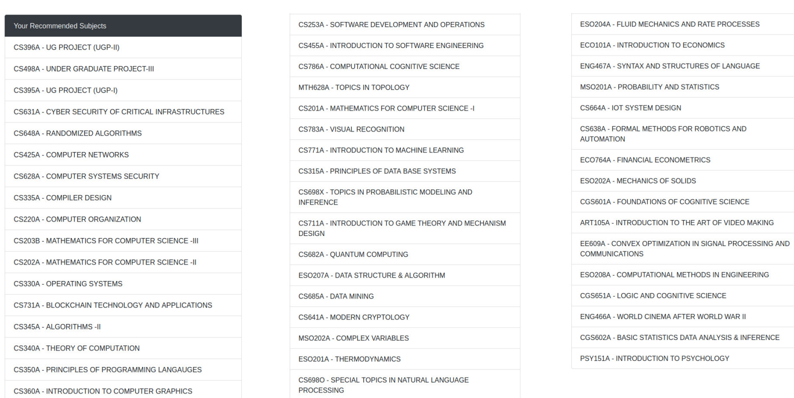

Figure 4: Recommendation output