Course Recommendation System - IIITD

Shreyash Arya[2015097] and Sarthika Dhawan[2015170]

Indraprastha Institute of Information Technology, Delhi 110020, India

Abstract. This paper aims to develop a system that will help in recommendation of courses for an upcoming semester based on the performance of previous semesters.

Keywords: Grades dataset, Courses, Students, Recommender System, Collaborative Filtering

1 Introduction

It has always been a tough choice for the students to choose the courses in different semesters in which there is possibility to score good grades apart from the interest in the course. IIIT-Delhi offers variety of courses with mandatory courses in first 4 semesters (with exception of 2 to 3 electives) and all elective courses from fifth semester onwards. Hence, choosing the courses based on the verbal recommendation from the seniors, instructors and fellowmates becomes a hectic task. For easing this process of course recommendation for an upcoming semester, we have developed a system which deploys simple yet powerful recommendation techniques such as auto-encoders, hybrid matrix factorization and similarity based approaches. It is a GUI based system which takes an input of student's ID (which is stored in the backend database) and semester for which the student wants to get the recommendation. Then, it outputs the top 5 courses that the student can choose based on his/her performance in previous semesters. Also, a confidence score is provided for each recommended course.

2 Dataset

2.1 Description

The dataset has been acquired from the official IIIT-Delhi academics department for the students of 7 Computer Science passout batches. The dataset consists 739 students and 306 subjects with mapping of each student to the grades for each course the student has taken throughout the duration of their degree. The courses are spread over 8 semesters (also including the courses offered to student with extended semester). The fields in the dataset include Serial Number(SN), Roll Number(anonymized) of the student, Batch/Term Code in which the course was offered, Course Code, Course name, Credit offered for the course, Grade obtained by particular student in the course, SPI (Semester Percentile Index) - Average GPA(Grade Point Average) for the current semester and CPI (Cummulative Percentile Index) - Overall GPA.

2.2 Processing

The data is processed from CSV format to JSON format. Each student is mapped to all the courses with the grade and semester in which the course was offered. If the student has taken the course, then the course key(under the particular student key) will have the corresponding grade obtained in the course and the semester in which the course was offered. Each user and course is mapped to a unique integer ID. Incomplete courses, courses for which leave application was given by the student and courses with grade W(weak) are given grade score 0. Online courses with grade S are given grade score as 10 while with X are given grade score 0. For creation of train and test matrices we included only courses that were offered in first 8 semesters as any semester beyond that signifies the presence of backlog or extended semester. For such cases we took the maximum grade score provided to the student in a particular course including all extended semesters and backlogs. Every course has a list of semesters in which it was offered as the same course can be floated in more than 1 semester. Various split ratios were considered and 5 fold cross validation was done for creating the matrices. Test matrix only considered the student-course pair for courses in 5th, 6th 7th and 8th semesters.

2.3 Dataset Distribution

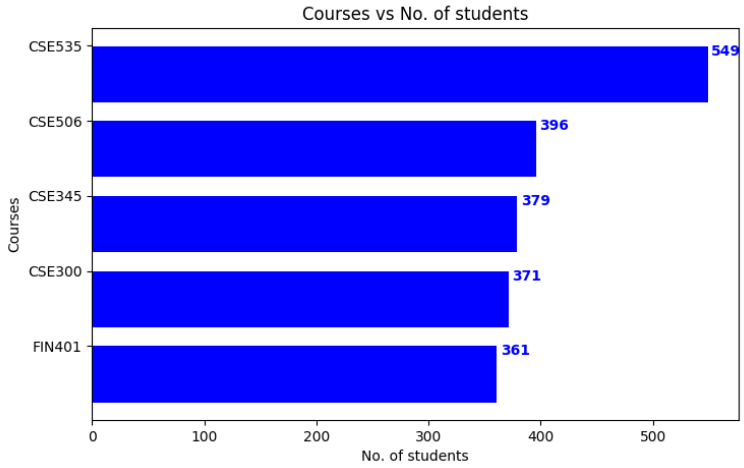
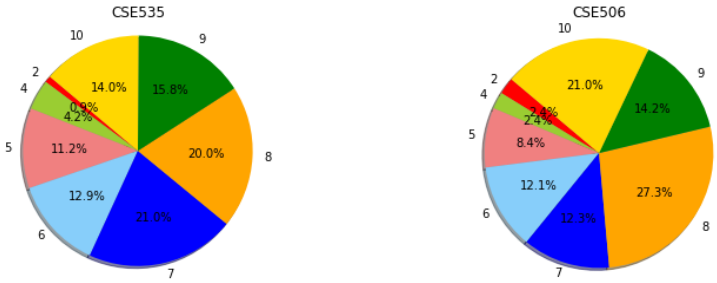


Fig. 1: Top 5 elective courses. (CSE535 - Mobile Computing, CSE506 - Data Mining, CSE345 - Foundation of Security, CSE300 - Software Engineering, FIN401 - Foundations of Finance)



CSE535 CSE506

Fig. 2: Grade distribution over the popular courses.

Student Distribution: Fig. 1 shows the most popular electives that are present in the dataset and have a high probability to be recommended as it has the most number of students and there is a good chance of finding similar students.

Grade Distribution: Fig. 2 represents the grading distribution over the popular courses and we can infer that most of the grades from distributed in the range of 10 to 7 and hence, have a high probability to be recommended (as most similar courses with top grades are recommended).

3 Methodology

This is a warm start problem as no useful metadata is available in the dataset that is relevant to the problem we are trying to tackle. Student vs Course matrix is created with grades(mapped from letter grade to numerical value) as the values in the matrix. This matrix is then fed into different recommendation algorithms as discussed below:

3.1 User-based and item-based recommendation approach using cosine similarity and k-nearest neighbours

This is one of the most commonly used method incorporated earlier in the task of predicting the missing ratings [1][2]. The basic strategy is to find the most similar students (in user-based method) or the courses (in item-based method) and try to give the grade scores based on highest similarity score. The similarity score is calculated using the cosine similarity.

$$cosine\_simimarity(v, w) = \frac{\mathbf{v}.\mathbf{w}}{|\mathbf{v}||\mathbf{w}|}$$

Also, the cases where there are no grades for the courses are handled.

Another modification done is to incorporate the top-k most similar users/items from the user-subject matrix and predict the grade scores based on these neighbour's grade scores.

User based grade prediction formula

$$p_{a,i} = \bar{r_a} + \frac{\sum_{u=1}^{n}(r_{u,i} - \bar{r_u}) * w1_{a,u}}{\sum_{u=1}^{n} w1_{a,u}}$$

Item based grade prediction formula

$$p_{a,i} = \frac{\sum_{k=1}^{n} r_{a,k} * w2_{i,k}}{\sum_{k=1}^{n} w2_{i,k}}$$

where p(x,y) is the predicted grade score of student x in course y, n is the number of neighbours, $r_{x,y}$ is the grade score of student x in course y, $w1_{x,y}$ is the similarity between students x and y and $w2_{x,y}$ is the similarity between courses x and y.

3.2 Hybrid Matrix Factorization

LightFM[5] is a python package that implements a number of popular recommendation algorithms for both implicit and explicit feedback, including efficient implementation of various ranking losses. Weighted Approximate-Rank Pairwise (WARP)(1) which maximises the rank of positive examples by repeatedly sampling negative examples until rank violating one is found and logistic loss(2) function optimizations were used with the traditional collaborative matrix factorization techniques which decompose the user-item interaction matrix into the product of two lower dimensionality rectangular matrices to predict the grades.

$$dU = L * (self.V[i] - self.V[j]) \tag{1}$$

where L is the number of trials required to find a violating negative column. U and V are factors , which are obtained during WSabie procedure[4]. V[i] - the sampled positive column V[j] - the sampled negative column.

$$J(w) = \sum_{i=1}^{m} y^{(i)} log\left(\frac{1}{1+e^{-w^T x}}\right) + (1 - y^{(i)}) log\left(1 - \frac{1}{1+e^{-w^T x}}\right) \tag{2}$$

where y(i) indicates the ith label in the training data.

3.3 Auto-encoders

We have used the item-based AutoRec model[1][6]. As shown in Figure 1 the model accounts for the fact that each rating is partially observed by only updating during backpropagation those weights that are associated with observed inputs. It also regularises the learned parameters so as to prevent overfitting

on the observed ratings. Students are analogous to users, courses to items and grades to ratings for our system.

$$min_\theta \sum ||r^{(i)} - h(r^{(i)};\theta)||_O^2 + \frac{\lambda}{2}.(||W||_F^2 + ||V||_F^2)$$

where $||.||_O^2$ is the contribution of observed ratings.

4 Results and Discussions

Table 1: Mean Absolute Error(MAE) for different methods incorporated on different split ratios for predicting the grades.

Method	**70-30**	**80-20**	**90-10**
User-based	1.76	1.74	1.75
Item-based	3.66	3.16	3.40
WARP MF	4.30	4.31	4.30
Logistic MF	2.26	2.30	2.32
Auto-encoders	**1.40**	**1.29**	**1.32**

Table 2: Root Mean Squared Error(RMSE) for different methods incorporated on differnt split ratios for predicting the grades.

Method	**70-30**	**80-20**	**90-10**
User-based	2.51	2.32	2.36
Item-based	4.19	3.60	3.86
WARP MF	8.87	8.88	8.84
Logistic MF	5.03	5.07	5.09
Auto-encoders	**2.45**	**2.30**	**2.32**

Table 3: Mean Absolute Error(MAE) for different methods incorporated on different folds for cross validation.

Folds	**User-based**	**Item-based**	**WARP MF**	**Logistic MF**	**Auto-encoders**
Fold 1	1.64	3.23	4.37	2.42	1.35
Fold 2	1.55	2.96	4.31	2.16	1.3
Fold 3	1.54	3.27	4.29	2.29	1.34
Fold 4	1.72	2.91	4.32	2.06	1.33
Fold 5	1.75	2.92	4.38	2.31	1.32
Average	1.64	3.058	4.334	2.248	**1.32**

Table 4: Root Mean Squared Error(RMSE) for different methods incorporated on different folds for cross validation.

Folds	**User-based**	**Item-based**	**WARP MF**	**Logistic MF**	**Auto-encoders**
Fold 1	2.24	3.82	8.98	5.31	2.18
Fold 2	2.37	3.34	8.82	4.72	2.09
Fold 3	2.05	3.58	8.80	4.92	2.18
Fold 4	2.41	3.29	8.90	4.58	2.25
Fold 5	2.53	3.35	9.04	5.05	2.32
Average	2.32	3.476	8.908	4.916	**2.20**

Different collaborative filtering techniques are used to predict grades. MAE and RMSE values of different methods are shown in Table 1,3 and Table 2,4 respectively. Auto encoder based approach outperforms the other models as it can learn a nonlinear latent representation through activation function g(.) as compared to traditional matrix factorization approaches that learn a linear latent representation. User based model is giving better results than item based model. This shows that prediction of grade of a course based on similarity of the student with other students is much more significant than the similarity of the course with the courses he/she has taken. This can be due to the fact that many students prefer to take a combination of courses spanning different domains over the duration of their degree. So the relation between courses is not much prominent. Since the average number of grades per course is much more than those per student; high variance in the number of student grades leads to less reliable prediction for user-based methods.

WARP loss works on the basis of penalizing the negative samples (drawn from the remaining set after fixing the positive sample) if the prediction for negative sample exceeds the prediction of the positive sample. It tries to perform gradient update to fix the rank violation else the loss function is updated.[7] But one thing to notice is that the model works best where there are distinct two class positive and negative class such as -1,1 but here in our problem there would be multiple classes for each grade value (2,4,5,6,7,8,9,10) and there is very large probability that the selected positive class has less predicted score as compared to other and lead to loss. Logistic loss works comparitively better as does not assign zero penalty to any points. Instead, functions that correctly classify points with high confidence are penalized less. This structure leads the logistic loss function to be sensitive to outliers in the data.

80:20 train to test split ratio is giving more accuracy as compared to 90:10 in terms of both MAE and RMSE which shows that the model is not overfitting. Variation over different folds also shows auto encoder based approach as a better model.

New courses recommended are having a high predicted grade score which signifies the fact that if the user had taken these courses his overall grade would have been better as compared to what he/she has got now.

5 The Developed System

We have developed an interface for displaying the results of our model. It recommends new courses to the user based on entered ID and semester. The Evaluation tab shows the predicted grade and the actual grade of the student in top 5 recommended courses while the Recommended courses tab shows new recommended courses whose ground truth is not available with us, i.e., the grade of that student in that particular course is unavailable as he/she has not taken the course according to the dataset.

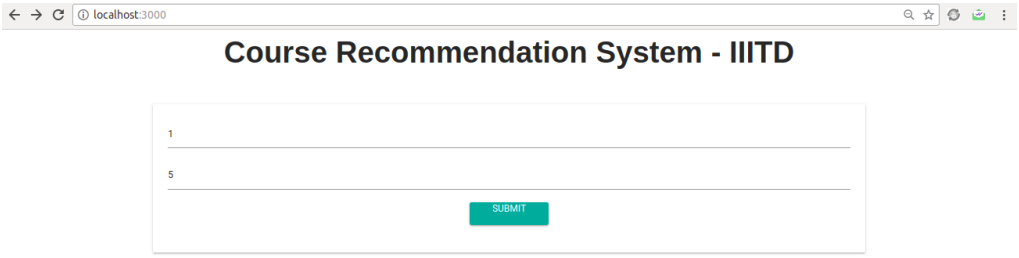


Fig. 3: Home Screen of the system.

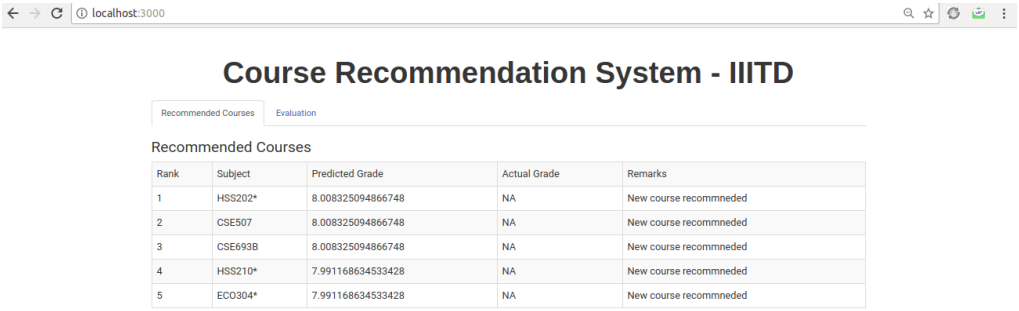


Fig. 4: Recommended Courses screen of the system.

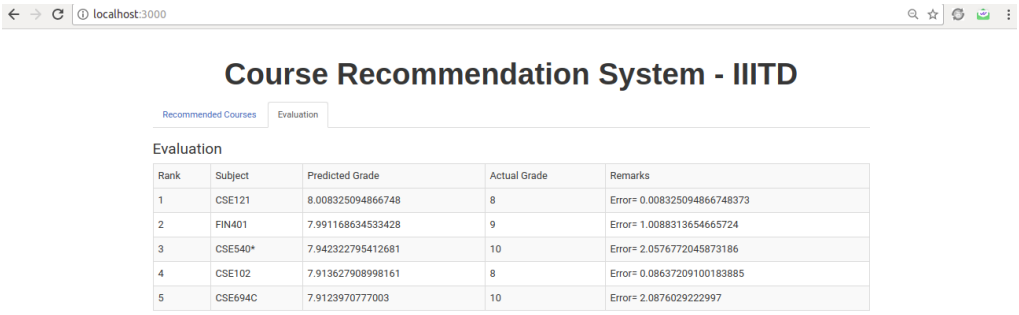


Fig. 5: Evaluation screen of the system.

6 Future Work

We were able to handle only warm start problems as the dataset did not contain appropriate metadata about students or courses. Dataset needs to be expanded to include several metadata features for better recommendation.

7 Supplementary Material

The project can be accessed at https://github.com/shrebox/cf-project.

References

1. Manos Papagelisa, Dimitris Plexousakisa: Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents. https://doi.org/10.1016/j.engappai.2005.06.010
2. Author, F., Author, S.: An Algorithmic Framework for Performing Collaborative Filtering.https://doi.org/10.1145/312624.312682
3. Author, F., Author, S.: AutoRec: Autoencoders Meet Collaborative Filtering. https://doi.org/10.1145/2740908.2742726
4. Weston, Jason, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to large vocabulary image annotation. IJCAI. Vol. 11. 2011.
5. LightFM Homepage, https://lyst.github.io/lightfm/docs/home.html.
6. AutoRec, https://github.com/HeXie-Tufts/Movie-Rating-Prediction-Autoencoder.
7. WARP loss, https://lyst.github.io/lightfm/docs/examples/warp_loss.html