ONLINE COURSE PERSONALIZED RECOMMENDATION SYSTEM

61431-FCS BSc (HONS) COMPUTING

BENEDICT, Nicholas 19086235D

Project Supervisor: Prof. LI Qing

Table of Contents

Table of Contents

Abstract

Deciding on which course to take during undergraduate study is important to gain relevant knowledge of certain professions (game developer, etc.) and maintain satisfactory Grade Point Average (GPA). Steps needed to get a clear understanding of a course is a tedious process, hence the idea of introducing a personalized recommendation system. The aim of this project is to create a content-based filtering recommendation system to provide courses that suits to the users'

preferences to help solve information overload issue and help reduce the steps needed to understand courses workloads, learning style, etc.

Background and Problem Statement

Registering elective courses for undergraduate students is one of the most important processes in their study. Each students have different aspirations (game developer, software engineer, fintech specialist, data scientist, etc.). To achieve their aspirations, they must take relevant courses to gain required knowledge to excel in their respective fields. Neglecting this process may cause students to spend their study credits on irrelevant courses and may affect Grade Point Average (GPA) negatively due to low course satisfaction. Hence students need to plan this process carefully and they need to consider the aspects of each course (is the course relevant to students' study plan, difficulty, workload, teacher's teaching style, is the material up to date, etc.). Students who care about their future career prospects and maintaining higher grades have several steps that can be taken to help them conclude which course to register for. However, there are some pitfalls within these options, so the idea of each university having their own Recommendation System (RecSys) would be undeniably beneficial to both students and lecturers alike by recommending courses relevant to students study plan (recommend game development-related classes and its pre-requisites for aspiring game developers).

The current situation and problems of known steps to the students' traditional decision-making process will be introduced as follows. Then the benefits of having a RecSys in higher-education institutes will be described. Continued with descriptions of several known techniques used in RecSys. Followed by the reason why Content-Based Filtering (CBF) method is picked for the system's development.

Introduction

When deciding on courses to register, Students may take several steps in considering whether an elective course is beneficial or not. They may ask their peers or academic advisors for advice on which courses to take based on their aspirations. However, there are several drawbacks to this approach as every student has different academic prowess and interests which could lead to unsatisfactory course experience based on the wrong information. Additionally, students who don't have acquaintances in the study field they are taking, especially international freshmen would have a hard time getting advice from their peers.

Students are advised by academic advisors to read the syllabus of each course to have a general idea of the course's workload, etc. Despite that, this solution doesn't provide insight on each lecturer's teaching style. Chetty et al [1] claimed that styles of teaching affects student's learning motivation which contributes to students' academic performance. With that in mind, conflicting learning and teaching style may result in lower GPA. Furthermore, Schmitt et al [2] argued that students lacking the ability to seek for information are prone to experience Information Overload (IO), where the students' mental capacity can't process large amount of information when concluding whether a course is worth taking, hindering their decision-making process. This phenomenon happens when students gather data from several sources (peer advice, syllabus reading, etc.). This phenomenon happens when students synthesize information from several sources, reflect on each impact generated by different courses, and to conclude the most beneficial decision [3].

Some universities implemented 'trial class', where students can take a course during the beginning of each semester so students can have a first-hand experience of teacher's teaching style and drop the course if students' expectations are not met. However, this approach is impractical due to limitations in actions students can do to adjust their courses, leaving students unable to freely try out different electives.

Lastly, students are encouraged to use a combination of these three approaches to understand each courses' learning and teaching aspects. Using this method offers a significant amount of course insight as each approach is used to cover up the other approaches' weaknesses. Nevertheless, this process is time-consuming and is prone to cause IO to the students. Which is why the idea of having RecSys is introduced.

Benefits of having RecSys

Personalized recommendation services provided by RecSys reduces the issue of IO by retrieving relevant course data in regards to the students' preferences [4]. With that in mind, RecSys may recommend courses relevant to their chosen interests and generate a study plan conforming to students' needs and preferences. Furthermore, each course will be given different rankings and alongside that, lecturers receive feedback from students that could help them to improve the course by modifying their teaching style, difficulty, workload, material relevance, etc. This ranking system may introduce healthy competition for growth in course satisfaction.

Types of Personalized RecSys

There are several main types of personalized RecSys in personalizing recommendations which are content-based filtering, collaborative-filtering, and hybrid filtering systems. The following paragraphs will describe each techniques merits and demerits. Continued with why content-based filtering is feasible for this use case.

Contend-Based Filtering (CBF)

Wang et al [5] described this type of RecSys as a system that compares data regarding users searched and rated item. It is important to note that only the user's data is used. It can be concluded that this type of system only recommends products or courses in students' cases based on their activities (search, taken courses, ratings of courses). Suppose student A searched and took a Software Engineering course, when the same student rated the course positively, they will be recommended to take the Software Project Management course due to some similarity in the course's contents. This approach's merits is that there is no need for other users' data to start recommending personalized courses to the original user [6], [7]. Moreover, CBF RecSys provides transparency by explaining how the user works and beneficial to new users [7]. On the other hand, this system's demerits are content overspecialization, where users only receives recommendations in regards to defined items in their user profiles and this type of RecSys' effectiveness relies heavily on available descriptive data [8].

Collaborative Filtering (CF)

This type of RecSys groups users with high similarities with other users into one "neighborhood" so that each user can receive recommended items that hasn't been rated by the user but has been reviewed positively by other users in their neighborhood [8]. It is inferred that this type's merits are being able to combat the issue of CBF's content overspecialization. Despite that, this approach brings up the cold-start problem, where such systems require large datasets on users explicit and implicit feedback first before it can make accurate recommendations [7]. It can be concluded that deploying CF RecSys is impractical for businesses with low number of recurring users. Using such technique would also cost time due to its complexity and testing required to ensure its accuracy.

Hybrid recommender system

Hybrid RecSys, as it names suggests, this system uses a combination of CBF and CF RecSys to provide recommendations with better accuracy than RecSys using only one filtering implementation

[6], [7]. This hybrid technique offsets each technique's weaknesses by implementing the other's strengths. For example, it is known that CF method suffers from cold-start problem and in this case, user profile will be built by using CBF approach to solve the problem. When the system has enough data to build a reliable user profile, CF method will be implemented to stop from over-specialization issue brought up by CBF.

Why CBF is chosen

Ideally, implementing a hybrid RecSys yields the most accuracy. However, higher-education institutions anonymize their course reviews data gathered from students bringing up the cold-start problem. For that reason, implementation of hybrid RecSys is impossible to do. This cold-start issue also applies to CF-based RecSys as well, hence why CBF is chosen over the alternatives. Unless students agree to have their data used in this RecSys, this approach will be used.

Objectives and Outcome

There are 3 tasks in this project. For task 1, it is imperative to create a website that can gather users' explicit (course ratings, comments, etc.) and implicit (items searched, time spent on looking at a course, number of clicks, etc.) feedback. Moreover, this website should be able to send request and receive responses from the API server. Task 2 is to create an API server that can receive requests and send responses to the front-end website application. Task 3 is to create a CBF RecSys to and implement the system into the API server. Data gathered from Kaggle will be used as a starting point to recommend courses to new users.

Methodology

In this section, there will be parts that will be described which are Front-end, Back-end, and RecSys development to implement a website app that is able to recommend relevant courses. The following paragraph will explain what steps are taken to realize this project.

Front-end development

Two main factors are considered before choosing a website application approach. First, accessibility of a website application is higher over mobile or desktop applications because website can be accessed by both smartphones and desktops. Secondly, I'm familiar with website application

development process using ReactJS, a front-end JavaScript framework to build reusable components. Because of this, development with familiar technologies will consume less time over learning mobile or desktop application technologies.

This website will be able to query for courses to the API server and visualize the data from the API to help the students' decision-making process. Furthermore, this website should be able to provide personalized recommendations by storing implicit data from the user into different sessions to ensure privacy. These data will be sent to the back-end server to get course recommendations.

Back-end development

Django, a python web framework will be used for this project to provide personalized recommendations. Django uses Python as its programming language, which allows the leverage of several existing Python libraries to provide the personalized recommendations. This API server will be able to receive requests and send response to the front-end website. Data gathered from Kaggle in form of '.csv' file extension will be stored into an SQL database so that if there are requests to rate a course, the API server knows where to store the new data. User data will not be stored into the database to keep privacy concerns from being an issue.

RecSys

This RecSys will be using CBF method to provide personalized course recommendations. Initially, newcomers to the website will not have any data, so only trending courses are recommended. However, when users start to interact with the website, the data to create a user profile will be stored within the web application as sessions and the API server will not store any of the data from the web application's request so users don't have to be concerned about their data being stored. To evaluate this system's performance, several metrics will be used in the next section.

Testing and Evaluation Plan

Test-Driven Development (TDD) will be adopted in the implementation of front-end and back-end systems as needed to ensure that code snippets with complex logics are working as expected. For the React-based website application, this app will be using Jest, a testing framework for writing unit tests. With this, it is easy to ensure that this app behaves as expected when user queries certain courses and tell their course preferences. On the other hand, the back-end server using Django

framework uses the built-in Python library module called ‘unittest’ to write unit tests which ensures the server can process requests and correct proper response to the React website application.

To evaluate RecSys, using several known metrics such as Mean Average Precision at K (MAP@K), to see how many times the RecSys provides correct recommendation out of any given cases. Mean Average Recall at K (MAR@K) can be used to check whether any courses are relevant to the users’ preferences. Combining both metrics will provide rates of false positives, and false negatives that is important to see the correctness of the RecSys.

Project Schedule

Date	Milestone
October 2021	- Project Plan Submission
November 2021	- Design user interface of the website application
November 2021 – December 2021	- Finalize features of Online Course RecSys - Do in-depth analysis of other works - Create prototype front-end website for displaying courses
3 January 2022	- Interim assessment submission
4 January 2022 – 10 April 2022	- Create API server for processing incoming requests - Implement CBF RecSys into the API server
April 2022	- Accuracy testing of RecSys system

Resources Estimation

Currently, it is estimated that training will take some reliable GPU to do calculations on the dataset to create an accurate model. With the help of open-source data from Kaggle, this project can be done within the Project schedule timeline.

References

[1] N. D. S. Chetty *et al.*, “Learning Styles and Teaching Styles Determine Students’ Academic Performances.,” *Int. J. Eval. Res. Educ.*, vol. 8, no. 4, pp. 610–615, 2019.

[2] J. B. Schmitt, C. A. Debbelt, and F. M. Schneider, “Too much information? Predictors of information overload in the context of online news exposure,” *Inf. Commun. Soc.*, vol. 21, no. 8, pp. 1151–1167, 2018.

[3] D. Dean and C. Webb, “Recovering from information overload,” 2011.

[4] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, “Recommender system application developments: A survey,” *Decis. Support Syst.*, vol. 74, pp. 12–32, Jun. 2015, doi: 10.1016/j.dss.2015.03.008.

[5] Y.-Y. Wang, A. Luse, A. M. Townsend, and B. E. Mennecke, “Understanding the moderating roles of types of recommender systems and products on customer behavioral intention to use recommender systems,” *Inf. Syst. E-Bus. Manag.*, vol. 13, no. 4, pp. 769–799, 2015.

[6] D. Das, L. Sahoo, and S. Datta, “A survey on recommendation system,” *Int. J. Comput. Appl.*, vol. 160, no. 7, 2017.

[7] P. B. Thorat, R. M. Goudar, and S. Barve, “Survey on collaborative filtering, content-based filtering and hybrid recommendation system,” *Int. J. Comput. Appl.*, vol. 110, no. 4, pp. 31–36, 2015.

[8] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, “Recommendation systems: Principles, methods and evaluation,” *Egypt. Inform. J.*, vol. 16, no. 3, pp. 261–273, Nov. 2015, doi: 10.1016/j.eij.2015.06.005.