Technologies and Algorithms for the Implementation of the Recommendation System for Creating an Individual Study Plan for a Higher Education Student

Olena Hlazunova and Yaroslav Ponzel

October 18, 2024

Technologies and algorithms for the implementation of the recommendation system for creating an individual study plan for a higher education student

Olena Hlazunova[1], Yaroslav Ponzel[2]

[1]*Doctor of Pedagogical Sciences, Professor, National University of Life and Environmental Sciences of Ukraine, 16a Heroiv Oborony Street, Kyiv, 03041, Ukraine*

[2]*Assistant of the Department of Information Systems and Technologies, National University of Life and Environmental Sciences of Ukraine, 16a Heroiv Oborony Street, Kyiv, 03041, Ukraine*

Abstract

The functionality of recommendation systems is implemented in a lot of popular services and marketing platforms, as they provide an opportunity to analyze and filter information in terms of choosing the most suitable products, content, or services for the user according to his preferences and capabilities. However, in the education, such functionality is also in great demand, especially when it comes to choosing educational content according to preferences and learning styles in order to create an educational trajectory based on previous experience and academic performance. Lack of understanding of their capabilities and preferences or inability to build a structured learning vector is inherent for applicants and first-year students. That is why this article analyzes the main problems in the implementation of a recommendation system for building an individual study plan for a higher education student, identifies the main ways to implement such a system, and builds a basic algorithm for a our recommendation system that can provide top disciplines for a student to study based on his or her previous learning outcomes, the learning outcomes of students of the same specialty for a certain period, and the factor of similarity of subjects and the choice of his or her specialization.

Keywords

Recommendation system, individual study plan, collaborative filtering, C#, Microsoft.ML

1. Introduction

According to the draft law No. 10177 "On the Development of Individual Educational Trajectories and Improvement of the Educational Process in Higher Education" [1], higher education students will be able to independently determine the format and trajectory of their studies. Students will have the opportunity to independently create the scope and timing of their studies, which will bring Ukrainian education closer to the European level. It will also provide the opportunity for

DigiTransfEd 2024: 3rd Workshop on Digital Transformation of Education, co-located with the 19th International Conference on ICT in Education, Research, and Industrial Applications (ICTERI 2024, September 23-27, 2027, Lviv, Ukraine

o-glazunova@nubip.edu.ua (O. Hlazunova); yaroslavponzel@nubip.edu.ua (Y. Ponzel)
https://docs.google.com/document/d/1QZJtcNKgptzB6jNwUSXkpZ3gKyij_vhxwLvE64nVg0s/edit#heading=h.cn7fagdotg7g (O. Hlazunova); https://docs.google.com/document/d/1TNfEZvbkxmKeO-AderFi77sufsFIlsITnywUAUW2fhs/edit#heading=h.a5qqzzkx8qm1 (Y. Ponzel)
0000-0002-0136-4936 (O. Hlazunova)

the students to individually build their learning trajectory by choosing the period of study of compulsory subjects and subjects that they choose within the curriculum of their specialty.

However, there are many problems that students face, including a lack of understanding of how to properly build their path to professionalism. As a result, the applicant may choose subjects that do not best suit his or her preferences and capabilities, or make a wrong choice of subject based on a small sample of data provided by friends or family.

Higher education institutions are also interested in the correct construction of the curriculum for their students, because a lot of important factors may directly depend on this, such as academic performance, satisfaction with the university, the qualification level of the specialist, and even the level of happiness of the person.

2. Problem statement

Based on the above, we are faced with the task of finding mechanisms and means to build an individual study plan for a higher education student, which must be resolved to meet the needs of both the educational institution and its students.

Let's identify the main problems that appear on the way to solving this problem:

- Scalability - of course, each student needs to build a personalized learning schedule. For example, about 26 thousand students study at the National University of Life and Environmental Sciences of Ukraine and its separate structural subdivisions [2], and although there were about 4162 freshmen [3]. And not only freshmen dream of building an individual curriculum, but also senior students want to have a constant review of the plan and a possible change in its vector. For such a volume of work, a higher education institution either needs to recruit a large staff to train, organize processes and communication with students, and constantly spend financial resources on the work of this structure, or it needs to automate processes and entrust it to a certain system.
- Big data – it is a problem that appears due to the need to process a large amount of data to provide quality advice to build an individualized plan for a student. This can be a list of subjects that are recommended for a certain kind of specialists, or a performance of students who studied in the same specialty before the student, or feedback from students who have passed subjects that can be recommended for the student, etc. It is better to entrust this work to an automated system than to a human, as this will help save time and resources needed to make a recommendation.
- Changing the calculation model - the problem appears when recommendations in creating a study vector for a student were built using one algorithm based on one data, and then there was a need to make the algorithm more accurate or expand its data cluster for processing. For an automated system, this will not be a problematic factor, and it will only require rewriting the program algorithm for reading and processing data. To make changes to the data processing algorithm in a system where a significant amount of work is performed by human resources, this will cause confusion, errors, the need to rewrite existing schedules, training to educate staff, etc.
- Accessibility - it is obvious that the automated system will be available for student requests at any time and in any quantity, which is impossible for a recommendation system based

on the employee-student model, as in this case it can only happen during working hours, in a limited number of requests and with the intervention of negative factors of influence, such as illness, a long distance between the consultant and the consultee, factors related to the war, etc.
- The human factor - it is a factor that is always present in a system where there is human influence, and the greater this influence, the greater the risk of error.

3. Review of recent works

Many researchers described how to implement recommendation systems. For example, in [4] researchers Jing Li and Zhou Ye first discovered that the development of personalized and effective recommendations for educational resources has become a hot point of research in modern educational platforms, after which, based on the collaborative filtering recommendation algorithm, they developed their system for providing recommendations for studying courses for a student. As noted in the article that recommendation system not only helps users to quickly find high-quality information that interests them, but also saves their time and costs.

In [5], Bhaskar Mondal, Om Patra, Sanket Mishra, and Priyadarshan Patra also propose an algorithm that uses collaborative filtering to be applied in a cluster to recommend several appropriate courses.

In article [6] authors present a software architecture aimed at developing learning assistant software that assists students in identifying, organizing, and achieving their personal educational goals based on their individual needs and interests. The system can be used in ILIAS or MOODLE learning support software and that is optimized for the environment of a joint research project at the universities of Bremen, Hanover and Osnabrück.

Also there is a different way to build your recommendation system, and Cem Dilmegani, principal analyst at AIMultiple, says in the article [7] that you can build your own solution with algorithm that more appropriate for you case, but you should do it when you are in a niche domain where recommendation engines were not used before or you own one of the world's largest marketplaces where slightly better recommendations can make an important difference in your business outcomes.

To predict students' future grades based on past grades, authors in the article [8] tried three different algorithms: a Linear Regression model based on least squares, a Random Forest Regressor with 100 decision trees, and an Artificial Neural Network with 4 dense layers and ReLU activation functions. All of them were implemented in their web application and could be used by students.

But after analyzing a large number of articles, we can conclude that there is no single correct way to build a recommender system. This is exactly what the authors' article [9] is about, where they analyzed a significant amount of material on recommendation systems and concluded that although hybrid strategies are the main development technique in the analyzed studies. However, it was noted that there is no universal general model or framework for educational choice recommendations, as each recommendation system must be specific to a particular context and type of data.

4. The purpose and task of the research

Taking into account the above-mentioned problems, we can conclude that the implementation of an automated system is the most priority way to provide personalized recommendations for students in building their curriculum. Therefore, the goal we set for ourselves is to identify technologies and develop algorithms that can be used to implement the basic logic of a recommendation system that will be able to solve the problem of forming individual educational plans based on processing large amounts of data.

5. Results of the research

There are two options for implementing a system for building an individual study plan:

1. Writing our own algorithm for reading and processing data from scratch, which could satisfy our problems. The advantage in this case is that we can fully customize the work of such a model to our needs, the disadvantage is the need to solve problems and tasks that have long been solved in the second method of implementing an automated system.
2. Using a recommendation system based on machine learning, which is a subtype of artificial intelligence that is widely used to predict the best choice of a particular product or service for a user, based on numerous factors, such as personal preferences of the user, preferences of his or her potential associates, or with the calculation of certain filters and restrictions.

The second method already contains a set of ready-made solutions that will help filter and process data and provide the most accurate forecast for the user. A recommendation system can be implemented in many frameworks and programming languages, including Python, Java, .NET, etc. That is why, when choosing a way to implement an automated system for building a curriculum for a higher education student, you should pay attention to this technology, its capabilities and its family of algorithms in general.

In article [10], a recommendation system is defined as an artificial intelligence algorithm, usually associated with machine learning, that uses a large amount of data to make a recommendation or guess for a particular user. Recommendation systems began their development as much as 40 years later [11] and gained such a powerful development to a greater extent to solve the problem of users in the corporate segment. Companies quickly realized that having such a system could be a big advantage over competitors, so they began to invest heavily in this area. For example, Netflix and YouTube, which is one of the most popular content viewing platforms as of 2024 [12], actively invested resources in the development of recommendation systems for their platforms in the first decade of the 21st century [13].

There are two general ways to generate candidates: content-based filtering and collaborative filtering [14]. The first type is able to make recommendations based on the personal preferences of the user, and the second type is able to make recommendations based on the preferences of a subgroup of users who are similar to the target user. Content-based filtering takes into account only the preferences of the student, which is not desirable for a recommendation system for a higher education institution, since although universities can train specialists in a narrow profile, possession of basic diverse knowledge in their specialized field is mandatory to create a highly qualified specialist. The experience of previous generations is also an important factor in choosing a future study plan, as students are likely to choose subjects that are most likely to be passed with the highest probability of success and have the best possible feedback from senior students. That's why collaborative filtering is the best fit for our requirements, as it is able to solve our target task of recommending subjects based on the student's own academic performance and the results of other students.

In addition, it is advisable to introduce a factor of similarity of subjects into this recommendation system. This will allow us to diversify the factors of finding the most optimal subject for a student and increase the accuracy of the forecast. It is often the case that subjects are either very similar or even identical in content, but they are entered into the database as separate unrelated entities. If we introduce the similarity factor into the model for the recommendation system, we can additionally tell it that these items are close to each other, so there is a need to analyze them as related.

In our study, we used matrix factorization, which is a simple model for displaying a discrete data set in a vector space and is used in joint filtering, on the basis of which a further forecast for the end user is built. The formula for matrix decomposition can be as follows:

$$R \approx UV^T$$

In this formula:

- (R) - rating matrix;
- (U) - matrix of users;
- (V^T) - transposed product matrix.

Matrix factorization has both advantages, such as the ability to provide a personalized forecast based on the interaction between users and products, and the ability to work with different types of data, but also has disadvantages, such as the mandatory need for a large amount of data to provide an accurate forecast, and the complexity of computation. This set of advantages and disadvantages is satisfactory for our task, since we need a personalized prediction based on the grades of the target student and other students, and a large amount of data will always be present in a system where many academic performance results are constantly recorded.

The algorithm of such a recommendation system is shown in (figure 1).

The main source of data with which our proposed algorithm works are the grades of a specific student during the period of study, the similarity of subjects and grades from the subjects of students of the same specialty for the last 3 years. Thus, the recommendations that we will receive after training the system will be relevant, as they are based on fresh data, and useful in building our future vector of individual training by profession and preferences.

To evaluate the accuracy of the model, the root mean square error (RMSE) is used, which measures the average square error between the predicted actual values and the actual values. The formula for the root mean square error is as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},$$

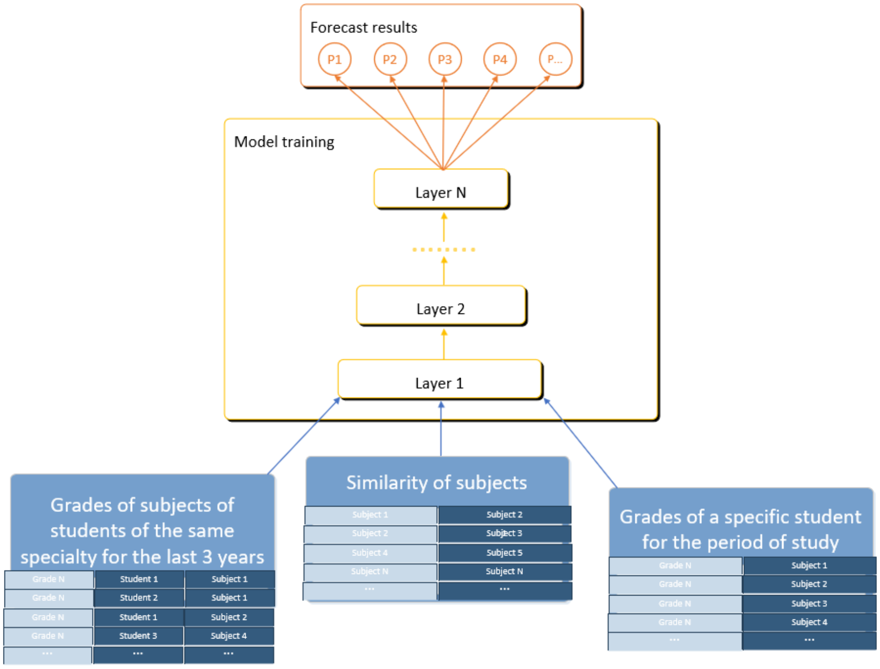

Figure 1: The basic algorithm of the recommendation system for building an individual study plan for a higher education student.

- (n) - count of observations;
- (y_i) - actual value;
- $(\hat{y}_i)$ - predicted value.

Using matrix factorization and methods for evaluating the accuracy of the assessment, the basic logic of the recommendation system was built in the C# programming language and using the Microsoft.ML library, which allows us to use machine learning. It includes:

1. To hook up our own libraries that will provide us with data. One (e.g. Models) to makes it possible to use our custom models in basic logic. Another one (e.g. Repositories), to provides an opportunity to extract custom specific data for training models.
2. To define the main models for machine data analysis (e.g. StudentCourse for training base data and CourseRecommendation for prediction result).
3. To enable client code functionality, it is essential to define a specific service (e.g. SubjectRecommender) that can be instantiated and utilized from any location. This service will operate based on the input of certain parameters. In this context, the unique student identifier will serve as the primary key. Utilizing this identifier, we will be able to locate and analyze data directly associated with the user and process it to generate relevant recommendations:
4. To build a model for learning data, we need to get our own custom students score data, identify the training data, and create a pipeline based on it. For this purpose, we got

custom information about the scores of students in disciplines of the same specialization as our target student. Then we converts specific custom information to ML training data. After that we split the dataset into the train set and test set according to the given fraction. And finally, we create matrix factorization trainer with advanced options, which predicts element values in a matrix using matrix factorization:

```
var studentDisciplinesScoresOfTheSameSpecialization =
    _studentDisciplineRepository
     .GetStudentDisciplinesScoresOfTheSameSpecialization(studentId);
var trainingData = mlContext.Data.LoadFromEnumerable(
    studentDisciplinesScoresOfTheSameSpecialization);
var dataSplit
   = mlContext.Data.TrainTestSplit(trainingData, testFraction: 0.2);
var matrixFactorizationTrainerOptions =
    new MatrixFactorizationTrainer.Options {
        MatrixColumnIndexColumnName = "StudentIdEncoded",
        MatrixRowIndexColumnName = "CourseIdEncoded",
        LabelColumnName = "Label", NumberOfIterations = 20,
        ApproximationRank = 100 };
var pipeline = mlContext.Transforms.Conversion
    .MapValueToKey(
     outputColumnName: "StudentIdEncoded", inputColumnName: "StudentId")
           .Append(mlContext.Transforms.Conversion.MapValueToKey(
     outputColumnName: "CourseIdEncoded", inputColumnName: "CourseId"))
        .Append(mlContext.Transforms.Text.FeaturizeText(outputColumnName:
           "CourseTypeFeaturized", inputColumnName: "CourseType"))
        .Append(mlContext.Recommendation().Trainers
          .MatrixFactorization(matrixFactorizationTrainerOptions));
```

5. To train machine learning model using the training dataset and make some predictions using trained model on some trained data. Then we evaluate the accuracy of the calculations using Root Mean Squared Error:

```
var model = pipeline.Fit(dataSplit.TrainSet);
var predictions = model.Transform(dataSplit.TestSet);
var metrics = mlContext.Regression.Evaluate(
   predictions, labelColumnName: "Label", scoreColumnName: "Score");
Console.WriteLine(
    $"Root Mean Squared Error: {metrics.RootMeanSquaredError}");
var predictionEngine = mlContext.Model.CreatePredictionEngine
    <StudentDiscipline, CourseRecommendation>(model);
```

6. And finally, with developed a mechanism for providing recommendations, we can provide a certain result of the algorithm's work for the student, based on a specific sample of

user data. For example, we can get for our calculation already passed disciplines by target student, disciplines that was passed by target student and all available not passed disciplines. Based on this data we can provide some prediction of top 20 courses that was not learnt by target student:

```
var studentPassedDisciplines =
    _studentDisciplineRepository.GetPassedDisciplines(studentId);
var studentSelectedDisciplines =
    _studentDisciplineRepository.GetSelectedDiscipline(studentId);
var notSelectedDisciplines =
    studentDisciplinesScoresOfTheSameSpecialization
      .Where(x => studentPassedDisciplines.All(y => y != x.CourseId)
           && studentSelectedDisciplines.All(y => y != x.CourseId))
         .GroupBy(x => new { x.CourseId, x.CourseType })
         .Select(x => x.First());
var courseInfoForPredicition = notSelectedDisciplines
   .Select(notSelectedDiscipline => (notSelectedDiscipline.CourseId,
        notSelectedDiscipline.CourseType)).ToList();
var predictionResults = new Dictionary<int, float>();
foreach (var (courseId, courseType) in courseInfoForPredicition) {
   var prediction = predictionEngine.Predict(new StudentDiscipline {
        StudentId = studentId,
        CourseId = courseId,
        CourseType = courseType
        });
   Console.WriteLine($@"Predicted preference for student {studentId}
       for course {courseId} ({courseType}): {prediction.Score}");
    predictionResults[courseId] = prediction.Score;
}
var topPredictedCourses = predictionResults
    .OrderByDescending(x => x.Value).Take(20).ToList();
foreach (var course in topPredictedCourses)
    Console.WriteLine($@"The course that suits you best and
       that we recommend you study: {course.Key}:{course.Value}");
}
```

The proposed basic logic of the recommendation system obtains custom specific information about the scores of students in disciplines of the same specialization as our target student. Next, the algorithm contains the main logic of the recommendation system, creates a matrix factorization trainer with advanced options that predicts element values in a matrix using matrix factorization, evaluates scored regression data, and uses the generated prediction engine to make several predictions about the top 20 courses that the target student have not passed yet. This described logic contains some custom classes and repositories for presentation purposes only and can be replaced with your own.

6. Conclusions

The article analyzes the task of building a curriculum that best suits the student's preferences and capabilities, identifies the main problems that arise on the way to solving such a task, and considers ways to overcome such difficulties. A basic algorithm has been built that will give a start to the work of a recommendation system, the purpose of which is to provide a student with recommendations of subjects to study, based on his or her previous learning outcomes, the learning outcomes of students of the same specialty for a certain period, and also on the factor of similarity of subjects. Based on this data and the proposed basic algorithm, the system can provide the top subjects for a student to study at a basic level. These subjects will be specialized, will be responsible for the next level of qualification of a young specialist, and will contain the highest probability of their successful completion. To implement such a system, it is proposed to use the ML.NET machine learning library for the C# programming language.

References

[1] The verkhovna rada of ukraine adopted the law on the development of individual educational trajectories and improvement of the educational process in higher education (2024). URL: https://www.rada.gov.ua/news/razom/248731.html.

[2] About the university (2023). URL: https://nubip.edu.ua/about.

[3] The academic council summarized the results of the 2023 admission campaign and made changes to the schedule of the educational process (2023). URL: https://nubip.edu.ua/node/134340.

[4] J. Li, Z. Ye, Course recommendations in online education based on collaborative filtering recommendation algorithm (2020). URL: https://onlinelibrary.wiley.com/doi/10.1155/2020/6619249.

[5] B. Mondal, O. Patra, S. Mishra, P. Patra, A course recommendation system based on grades (2020). URL: https://ieeexplore.ieee.org/abstract/document/9132845.

[6] F. Weber, J. Schrumpf, N. Dettmer, T. Thelen, A web-based recommendation system for higher education: Siddata (2022). URL: https://online-journals.org/index.php/i-jet/article/view/31887.

[7] C. Dilmegani, Recommendation systems: Applications and examples in 2024 (2024). URL: https://research.aimultiple.com/recommendation-system/.

[8] H. Grebla, C. V. Rusu, A. Sterca, D. Bufnea, V. Niculescu, Recommendation system for student academic progress (2022). URL: https://www.scitepress.org/Papers/2022/108163/108163.pdf.

[9] N. Kamal, F. Sarkar, A. Rahman, S. Hossain, K. A. Mamun, Recommender system in academic choices of higher education: A systematic review (2024). URL: https://www.researchgate.net/publication/378458769_Recommender_System_in_Academic_Choices_of_Higher_Education_A_Systematic_Review.

[10] Recommendation system (2024). URL: https://www.nvidia.com/en-us/glossary/recommendation-system/.

[11] Z. Dong, Z. Wang, J. Xu, R. Tang, J. Wen, A brief history of recommender systems (2022). URL: https://arxiv.org/pdf/2209.01860.
[12] G. Savinov, 10 most popular subscription streaming services (2024). URL: https://drukarnia.com.ua/articles/10-naipopulyarnishikh-strimingovikh-servisiv-za-pidpiskoyu-xvhE_.
[13] N. N. Qomariyah, Definition and history of recommender systems (2020). URL: https://international.binus.ac.id/computer-science/2020/11/03/definition-and-history-of-recommender-systems/.
[14] Candidate generation overview (2024). URL: https://developers.google.com/machine-learning/recommendation/overview/candidate-generation.