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

Olena H. Hlazunova[1], Yaroslav Y. Ponzel[1]

[1]*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 many popular services and marketing platforms, as they provide an opportunity to analyze and filter information in choosing the most suitable products, content, or services for the user according to his preferences and capabilities. However, in education, such functionality is also in great demand, especially when choosing educational content according to preferences and learning styles 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 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 speciality 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 draft law No. 10177, "On the Development of Individual Educational Trajectories and Improvement of the Educational Process in Higher Education" [1], higher education students can 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 allow the students to individually build their learning trajectory by choosing the period of study of compulsory subjects and subjects they choose within the curriculum of their speciality.

However, students face many problems, 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 many 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 finding mechanisms and means to build an individual study plan for a higher education student, which must be resolved to meet the educational institution's and its

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, 2024, Lviv, Ukraine

o-glazunova@nubip.edu.ua (O. H. Hlazunova); yaroslavponzel@nubip.edu.ua (Y. Y. Ponzel)
0000-0002-0136-4936 (O. H. Hlazunova)

students' needs.

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

- Scalability – 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]. However, there were about 4162 first-year students [3]. Not only do first-year students 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 particular system.
- Big data – 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 recommended for a certain kind of specialist, a performance of students who studied in the same speciality 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 appeared when recommendations in creating a study vector for a student were built using one algorithm based on one data. Then, the algorithm needed to be more accurate or expand its data cluster for processing. This will not be a problematic factor for an automated system, and it will only require rewriting the program algorithm for reading and processing data. Changing the data processing algorithm in a system where human resources perform significant work will cause confusion, errors, the need to rewrite existing schedules, training to educate staff, etc.
- Accessibility – 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 more significant this influence, the greater the risk of error.

3. Review of recent works

Many researchers described how to implement recommendation systems. For example, in Li and Ye [4] first discovered that the development of personalized and practical recommendations for educational resources had 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, the recommendation system helps users quickly find high-quality information that interests them and saves them time and costs.

Mondal et al. [5] also propose an algorithm that uses collaborative filtering to be applied in a cluster to recommend several appropriate courses.

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

Also, there is a different way to build your recommendation system. Cem Dilmegani, the principal analyst at AIMultiple, says in [7] that you can build your solution with an algorithm more appropriate for your case. However, you should do it when you are in a niche domain where recommendation engines have not been used before or you own one of the world's largest marketplaces where slightly better recommendations can make an essential 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 four dense layers and ReLU activation functions. All of them were implemented in their web application and could be used by students.

However, after analyzing many articles[5, 6, 7, 9], we can conclude that there is no single correct way to build a recommender system. This is precisely what the authors' article [10] is about, where they analyzed a significant amount of material on recommendation systems and concluded that hybrid strategies are the primary 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

Considering the abovementioned problems, we can conclude that implementing an automated system is the most crucial way to provide personalized student recommendations 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 algorithm for reading and processing data from scratch could satisfy our problems. In this case, the advantage is that we can fully customize the work of such a model to our needs. However, the disadvantage is the need to solve problems and tasks that have long been solved using 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 specific 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 [11], 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 [12] 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 significant advantage over competitors, so they began to invest heavily in this area. For example, Netflix and YouTube, one of the most popular content viewing platforms as of 2024 [13], actively invested resources in developing recommendation systems for their platforms in the first decade of the 21st century [14].

There are two general ways to generate candidates: content-based filtering and collaborative filtering [15]. The first type can make recommendations based on the user's preferences, and the second type can make recommendations based on the preferences of a subgroup of users who are similar to the target user. Content-based filtering considers only the student's preferences, which is not desirable for a recommendation system for a higher education institution since, although universities can train specialists in a narrow profile, possessing basic diverse knowledge in their specialized field is mandatory to create a highly qualified specialist. The experience of previous generations is also an essential 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 is why collaborative filtering is the best fit for our requirements, as it can solve our target task of recommending subjects based on the student's 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 that determine the most optimal subject for a student and increase the accuracy of the forecast. Subjects are often very similar or even identical in content, but they are entered into the database as separate, unrelated entities. Suppose we introduce the similarity factor into the model for the recommendation system. In that case, we can also tell that these items are close to each other, so it is necessary to analyze them as related.

Our study used matrix factorization, a simple model for displaying a discrete data set in a vector space. It is used in joint filtering, based on 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 data sources 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 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 model's accuracy, 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:

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

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

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

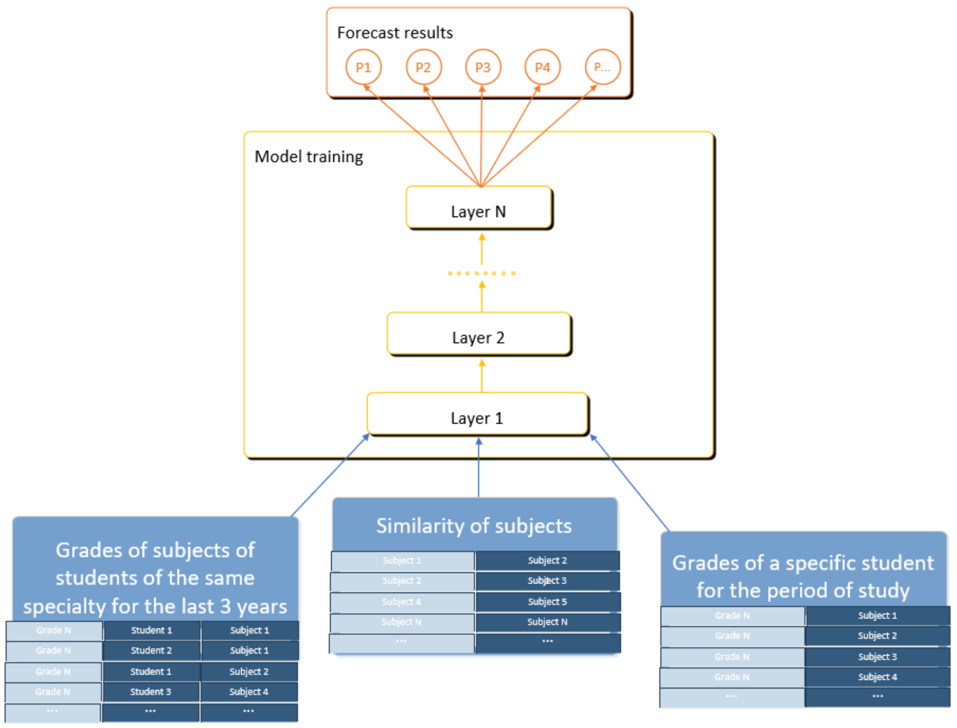

Figure 1: The basic algorithm of the recommendation system for building an individual study plan for a higher education student.

1. To hook up our libraries that will provide us with data. One (e.g. Models) to make it possible to use our custom models in basic logic. Another one (e.g., Repositories) allows one 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 specific parameters. The unique student identifier will serve as the primary key in this context. 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 custom students to 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 convert 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. Moreover, finally, we create a 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. Moreover, by developing a mechanism for providing recommendations, we can provide an inevitable 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 the target student, disciplines that the target student passed and all available not passed disciplines. Based on this data, we can provide some predictions of the top 20 courses that were target student did not learn:

```
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 with 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 custom classes and repositories for presentation 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 when 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 speciality 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, Complexity 2020 (2020) 6619249. doi:10.1155/2020/6619249.

[5] B. Mondal, O. Patra, S. Mishra, P. Patra, A course recommendation system based on grades (2020) 1–5. doi:10.1109/ICCSEA49143.2020.9132845.

[6] F. Weber, J. Schrumpf, N. Dettmer, T. Thelen, A Web-Based Recommendation System for Higher Education: SIDDATA: History, Architecture and Future of a Digital Data-Driven Study Assistant, International Journal of Emerging Technologies in Learning (iJET) 17 (2022) 246–254. URL: https://online-journals.org/index.php/i-jet/article/view/31887. doi:10.3991/ijet.v17i22.31887.

[7] C. Dilmegani, Recommendation Systems: Applications and Examples in 2024, 2024. URL: https://research.aimultiple.com/recommendation-system/.

[8] H. Greblă., C. V. Rusu., A. Sterca., D. Bufnea., V. Niculescu., Recommendation System for Student Academic Progress (2022) 285–292. doi:10.5220/0010816300003116.

[9] T. A. Vakaliuk, S. I. Pochtoviuk, Analysis of tools for the development of augmented reality technologies, in: S. H. Lytvynova, S. O. Semerikov (Eds.), Proceedings of the 4th International Workshop on Augmented Reality in Education (AREdu 2021), Kryvyi Rih, Ukraine, May 11, 2021, volume 2898 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2021, pp. 119–130. URL: https://ceur-ws.org/Vol-2898/paper06.pdf.

[10] N. Kamal, F. Sarker, A. Rahman, S. Hossain, K. A. Mamun, Recommender System in Academic Choices of Higher Education: A Systematic Review, IEEE Access 12 (2024) 35475–35501. doi:10.1109/ACCESS.2024.3368058.

[11] Recommendation System, 2024. URL: https://www.nvidia.com/en-us/glossary/recommendation-system/.

[12] Z. Dong, Z. Wang, J. Xu, R. Tang, J. Wen, A Brief History of Recommender Systems, 2022. URL: https://arxiv.org/abs/2209.01860. arXiv:2209.01860.

[13] G. Savinov, 10 Most Popular Subscription Streaming Services, 2024. URL: https://drukarnia.com.ua/articles/10-naipopulyarnishikh-strimingovikh-servisiv-za-pidpiskoyu-xvhE_.

[14] 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/.

[15] Candidate Generation Overview, 2024. URL: https://developers.google.com/machine-learning/recommendation/overview/candidate-generation.