Enhancing Online Learning

Newin Yamaguchi

1Affiliation not available

April 02, 2024

Abstract

In the contemporary digital landscape, online learning has emerged as a widely embraced mode of education due to its accessibility and cost-effectiveness. Timely delivery of courses tailored to learners’ needs is pivotal in maintaining their focus and commitment to learning. This project highlights the significant role of recommendation systems in attracting learners to focus on their learning paths and achieve their goals. It underscores the economic accessibility and the opportunities for individuals to engage with essential career skills. These concerns are particularly salient given the high cost of traditional education and the perceived elitism of certain institutions, which create barriers for economically disadvantaged learners.

The project employs a multi-faceted approach to enhance recommendation accuracy. Initially, it collects user information to gauge Feature Ratings, reflecting learners’ reactions to courses. Additionally, course descriptions undergo analysis to ascertain word importance and inter-course relevance. While a standalone content-based approach proves insufficient, the project adopts a collaborative filtering approach next. Here, Feature Ratings are learned through a K-Nearest Neighbors (KNN) model, leveraging a similarity metric to identify similar learners. Crucially, the project integrates these methodologies into a Hybrid Filtering model, combining the strengths of both approaches for optimal performance. Performance evaluation showcases the system’s efficacy, measured through Hit Rate and F1 Score accuracies, demonstrating its effectiveness in enhancing learning outcomes.

1. INTRODUCTION

The obstacles of online learning become the main issue on e-Learning platform. The primary aim of this project is to address these challenges by employing various feedback models to recommend courses tailored to individual user interests [1]. Many e-Learning platforms lack sophisticated algorithms for suggesting courses based on learners’ preferences [2]. Consequently, users frequently resort to selecting courses prominently displayed on the homepage, leading to a detrimental impact on study intention. The objectives of the project are twofold: firstly, to offer personalized course recommendations tailored to individual learners, and secondly, to systematically gather and analyse user data with the aim of enhancing the quality and accuracy of course recommendations over time.

The project scope involves identifying and utilizing appropriate algorithms based on relevant methodologies to develop a recommendation system. Clear and verifiable success criteria will be defined, aligned with the chosen methodologies. Necessary libraries for implementation will be specified, and a suitable dataset will be selected for experimentation. Our recommendation system aims to enhance user experience by predicting ratings for specific items based on individual preferences. We will utilize content-based filtering, analysing user features and preferences to recommend items, and collaborative filtering, leveraging past user interactions to personalized recommendations [3].

Our research is fortified by a robust framework comprising essential Python libraries such as Scikit-learn, SciPy, NumPy, and Pandas. These libraries form the backbone of our methodology, empowering us to
conduct predictive data analysis, solve complex mathematical equations, handle scientific computing tasks, and perform agile data analysis crucial for our research objectives.

With a focus on personalized course recommendations and user data analysis, our research endeavours to achieve two primary outcomes. Firstly, we anticipate a significant enhancement in user experience and satisfaction levels. By leveraging advanced recommendation algorithms, we aim to provide tailored course suggestions that align with individual preferences, ultimately fostering a more engaging and rewarding learning experience for users. Secondly, our research aims to enhance content discovery mechanisms, ensuring that users can easily explore and access relevant and valuable course content.

To achieve these objectives, we look forward to Kaggle as a trusted source of high-quality datasets essential for building and evaluating AI models. Kaggle not only provides access to diverse datasets but also facilitates data exploration, analysis, and collaboration, thereby enriching our research with valuable insights and fostering a collaborative environment for data-driven discoveries. Moreover, we also retrieve some datasets from an educational institution within Chiang Mai University. This research aims to improve decision-making, educational model, and technical way [4] to benefit society by enhancing health, safety, legal compliance, and cultural inclusivity.

2. RELATED STUDIES

To understand a whole picture of our recommendation, it is vital to thoroughly look for an overview of e-Learning course recommendations, focusing on the application of machine learning techniques [5]. It draws upon relevant research articles to establish a comprehensive understanding of the topic, aiming to provide insight into the complexities of these systems and their relevance in the field of machine learning.

Focusing on an e-Learning recommendation system that assists learners in accessing diverse topics without requiring personal information. The core functionality of the recommendation algorithm facilitates learners’ engagement with various online courses [7]. The implementation incorporates different techniques to address challenges, including the utilization of collaborative filtering [8]. The economic challenges faced by many individuals, especially during the COVID pandemic, highlight the importance of affordable education [9]. Developers should prioritize key components such as datasets, explicit and implicit ratings, learner behavior data, and results when implementing recommendation mechanisms [10].

2.1 Problems

The primary challenge in e-Learning course recommendation systems is the inefficient presentation of products to users, requiring them to make decisions to discern their needs or preferences. To address this challenge, it is essential to consider the pros and cons of different approaches and their practical behaviour [6].

2.2 Content-Based Filtering

The first step is the Feature Rating, where features describe users and items in a recommendation system [11]. This method works best when all items share the same set of features. However, it’s not suitable for items with different features. Compatibility across items’ features must be considered. In this case, content-based filtering assesses the relevance of courses based on their descriptions. Term Frequency and Inverse Document Frequency (TF-IDF) [12] are implemented to calculate the weights of courses’ descriptions.

Equation 1: Term Frequency

\[
tf(\text{term}, \text{document}) = \frac{f(\text{term}, \text{document})}{\sum_{\text{term}' \in \text{document}} f(\text{term}', \text{document})}
\]
Content-based filtering in recommendation systems offers advantages such as quick processing tailored to individual users and the ability to capture niche interests effectively. However, its reliance on feature definition and user knowledge can impact recommendation quality, and it is limited to suggesting items within a user’s current interests without the ability to expand those interests. Therefore, the collaborative approach is proposed.

2.3 Collaborative Filtering

Collaborative filtering recommends items based on the ratings of similar users. K-Nearest Neighbors (KNN) [13, 14] is employed for this purpose, utilizing a similarity metric to identify the most similar users. In addition to traditional collaborative filtering, the system also incorporates an item-based algorithm that analyses users’ behaviour records to calculate item similarities. This approach filters out item sets with high scores among those generated by the target user and utilizes the item similarity matrix to identify other items most like each item in the collection. By sorting and filtering items selected by the target user, the system then recommends the remaining items to enhance the recommendation accuracy and user satisfaction [15]. K-Nearest Neighbors (KNN) is a recommendation technique that uses a user-item matrix to calculate cosine distances between courses and then selects the top similar courses based on these distances. K-Nearest Neighbors (KNN) in recommendation systems has advantages such as not requiring explicit feature definitions and the ability to recommend diverse items without feature specification. However, it faces drawbacks like delayed recommendations for new items until users rate them, decreased accuracy in sparse matrices, and a tendency to prioritize popular courses over less-attended ones.

One common thread in recommender systems is the need to combine recommendation techniques to achieve peak performance [16]. All the known recommendation techniques in different ways. Hybrid recommendation systems [17] combine multiple techniques to enhance performance by customizing recommendations based on specific conditions and dataset requirements.

3. EVALUATION

Evaluation helps you assess how well your recommendation system is performing. It allows you to measure the accuracy of the recommendations made by the model and understand its effectiveness in suggesting relevant courses to users [18].

3.1 Training and Testing

Training and testing allow you to evaluate the performance of your recommendation system. By training the model on a subset of data and testing it on another subset that it hasn’t seen during training, you can assess how well the model generalizes to new data [19]. This helps in understanding if the recommendations made by the system are accurate and relevant.

3.2 Hit Rate

The performance of models is evaluated by predicting a certain number of courses for each user in the training dataset. Then, for each user in the test dataset, it will check if any of the courses they took are among the N predicted courses. If at least one of the predicted courses matches a course taken by the user in the test dataset, it is considered as a Hit [20].

3.3 F1 Score
This is a metric used to evaluate the performance of a classification model. The precision, recall, and F1 value are used as evaluation indexes to measure the performance of each algorithm [21]. The precision refers to the probability that the user is interested in the recommended course list, and recall refers to the probability that the course that the user is interested in appears in the recommendation list [22].

4. PROJECT STRUCTURE

Overall, the methodologies involved data preprocessing, implementing two different recommendation algorithms (TF-IDF and KNN), combining their results into a hybrid approach, and integrating the system to become a part of backend system behind the e-Learning platform for user interaction and visualization of results. Here’s a step by-step summary of the experimentation and results:

4.1 Data Cleaning

To prepare user and item datasets for analysis, several preprocessing steps are required. First, load the datasets in a format compatible with the specified structure. For the user dataset, reformat dates to a standardized format, standardize payment statuses and education levels for consistency, clean addresses to remove any inconsistencies or errors, and extract email domains from email addresses for further analysis if needed. Additionally, rename all columns in the user dataset for clarity and uniformity. In the item dataset, rename the course and description columns to enhance readability and understanding. Finally, after completing these preprocessing steps, save the modified datasets to new files for future use in analyses or recommendation system implementations.

4.2 Course Recommendation using TF-IDF and Linear Kernel

To perform course recommendation using TF-IDF (Term Frequency-Inverse Document Frequency) and a Linear Kernel approach, several steps are involved. First, the cleaned dataset containing user and item information is loaded. Then, the courses taken by a specific user are matched between the user dataset and the item dataset to retrieve their descriptions. Using TF-IDF vectorization, the weights of words in these course descriptions are calculated, giving higher importance to words that are rare across all courses but frequent within a specific course description. Next, a Linear Kernel is applied to compute the cosine similarities between all courses based on their TF-IDF vectors, measuring the degree of similarity between course descriptions. Finally, the top N recommended courses are returned based on their cosine similarity scores, providing personalized recommendations for the user based on their previous course choices. This approach leverages text analysis techniques and machine learning algorithms to generate relevant and tailored course suggestions.

In this example, we employ the Term Frequency-Inverse Document Frequency (TF-IDF) technique to assess word relevance in course descriptions across multiple documents. Initially, we calculate the Term Frequency (TF) for each word within a document, indicating how frequently a word appears in that specific document. For instance, in the course description "The programming language is difficult, The local language is easy," we compute TF values for words like "The," "programming," "language," "is," "difficult," "local," and "easy."

Next, we determine the Inverse Document Frequency (IDF) for each word, which evaluates the significance of a word by considering its frequency across all documents. Common terms like "The" and "professor" have lower IDF scores due to their prevalence across many documents, while specific terms like "web" or "network" have higher IDF scores due to their uniqueness. We combine the TF and IDF values using the TF-IDF formula (TF-IDF = TF * IDF) to gauge the relevance of a word within a specific document compared to its importance across all documents. For example, the TF-IDF score for the word "language" in Document 1 is calculated by multiplying its TF (1.3) with its IDF (approximately 0.18), resulting in a TF-IDF score of around 0.23. This process helps identify words that are both frequently occurring in a document and distinctive across the entire dataset, aiding in the determination of key terms for course recommendation purposes.
Following the TF-IDF computation, we move to cosine similarity. Cosine similarity measures the similarity between two vectors by calculating the cosine of the angle between them. In the context of course recommendation, we construct a user-item matrix where each row represents a course and each column represents a word, with values indicating TF-IDF scores. We then compute the dot product between each pair of course vectors to obtain a course-course similarity matrix. This matrix, filled with cosine similarity values ranging from 0 to 1, reflects the similarity between courses based on their descriptions. A higher cosine similarity implies greater similarity between courses, enabling us to recommend relevant courses to users based on a given course’s similarity to others in the dataset. For instance, a cosine similarity matrix like the one shown (Table 3.6: Item-Item Matrix) allows us to identify courses with similar content or themes, enhancing the precision of course recommendations in educational platforms or e-learning systems.

4.2 Course Recommendation using Feature Ratings and KNN

To implement a course recommendation system, the first step is to load the cleaned dataset containing information about users and courses. Next, feature ratings for each course given by users are determined and calculated. These ratings reflect user preferences and help quantify the level of interest users have in different courses. Following this, the model is trained using the user-course data to learn the relationships between courses based on user preferences. This training process enables the system to understand patterns and correlations in user behaviour, enhancing its ability to make accurate recommendations. Once the model is trained, it can predict and rank the distances of courses based on the courses that the user has taken, using techniques such as collaborative filtering or content-based filtering. Finally, the system returns the top N recommended courses for the given user, leveraging the learned relationships and user preferences to provide personalized and relevant course suggestions.

4.2.1 Feature Ratings Calculation

The feature ratings are also considered to recommend courses to individual users. The principle underlying this approach is to calculate the rating scores for each course based on the user’s historical background. The measurement of impressive score is calculated by observing the behaviours of users in each column as follows. Firstly, for the Email Column, users receive points based on the type of email domain provided: 0 points for no email, 1 point for a common email domain, and 2 points for an educational email domain. The Age-Education Column assigns points based on the user’s status: 0 points for being in the educational system, 1 point for just graduating, and 2 points for being in the working age. The Time Column awards points based on the suitability of the registration time for learning: 0 points for an unsuitable time, and 1 point for a suitable time. The Payment Column assigns points based on payment status: 0 points for overdue payment, 1 point for unapproved payment, and 2 points for on-time payment. Finally, the Address Column gives 1 point if the address is filled and 0 points if it’s blank. This scoring system helps assess various aspects of user information to categorize and prioritize users based on their readiness and suitability for engaging in learning activities or accessing certain services.

Furthermore, find the impressive level individually by considering the historical background of the user and the course, remove the feature rating columns to remain only user, course, and score, pivot the table to get the user-course matrix.

4.2.2 Nearest Neighbors Model

The process described involves fitting a k-nearest neighbors (KNN) model to a dataset where yellow, red, and blue colours represent three different courses. In this context, the KNN model is used to calculate the cosine distance between the red course and all other courses in the dataset. The cosine distance is a measure of similarity between two vectors, in this case, representing the features or characteristics of each course. After calculating the cosine distances for the red course with respect to all other courses, the next step is to select the top-most similar courses based on these distances. The KNN model identifies the nearest neighbors (courses) to the red course by ranking the cosine distances, where smaller distances indicate higher similarity. The topmost similar courses are then determined based on a predefined number of neighbors (k) specified in the KNN model.
Overall, this process leverages the KNN algorithm to assess the similarity between courses using cosine distances, enabling the identification of courses that are most closely related or like the red course based on their features or characteristics.

4.3 Hybrid Course Recommendation

This process outlined involves creating a hybrid recommendation system by combining results from two distinct approaches: TF-IDF and KNN. Firstly, the cleaned dataset is loaded, containing information about users and courses. The TF-IDF approach is utilized to analyse the relevance of words in course descriptions and calculate the importance of each word in recommending courses to users. Simultaneously, the KNN approach is employed to determine the similarity between courses based on user preferences and past interactions.

In the hybrid recommendation system, the results from both TF-IDF and KNN approaches are combined to enhance the accuracy and effectiveness of course recommendations. This integration allows for a more comprehensive analysis of course similarities and user preferences, leading to more personalized and relevant recommendations. Finally, the system returns the top N recommended courses for the given user by leveraging insights from both TF-IDF and KNN approaches, providing a holistic and improved recommendation experience.

To combine TF-IDF and KNN in a hybrid recommendation system, several steps are crucial. Firstly, we need to set weights and normalize the matrices of TF-IDF and KNN to ensure their compatibility and meaningful integration. This step involves assigning appropriate weights to the TF-IDF and KNN results based on their relative importance in the recommendation process. Normalizing the matrices ensures that their values are scaled appropriately for accurate comparison and combination. Next, the normalized matrices from TF-IDF and KNN are stacked in sequence horizontally to create a unified dataset that captures both the semantic similarities derived from TF-IDF and the user-item interactions modelled by KNN. This stacked matrix represents a comprehensive view of course similarities and user preferences, incorporating insights from both approaches. Finally, we apply the Nearest Neighbors algorithm to the stacked matrix to generate the final recommendations. This process is like traditional KNN, where the algorithm identifies the nearest neighbors to a given course based on the stacked matrix, ranking them by their similarity scores. The top N recommended courses are then returned to the user, leveraging the combined strengths of TF-IDF and KNN for more accurate and personalized recommendations.

4.3 Evaluation

The outlined evaluation process provides a structured approach for developers and researchers to measure and enhance the effectiveness of recommendation systems in delivering personalized and relevant recommendations to users. Firstly, the evaluation considers users who have engaged with multiple courses, reflecting a realistic scenario where users have diverse interests and preferences. Next, the training and testing datasets are divided based on a specified ratio, ensuring a balanced representation of user interactions across courses in both datasets. This division allows the model to familiarize itself with the training dataset, minimizing bias and ensuring that the model learns from a diverse set of user behaviours. Once trained, the model predicts courses for all users in the dataset, leveraging the learned patterns and relationships to generate recommendations. The accuracy of the recommendations is then calculated by measuring the similarities between the predicted courses and the actual test courses, providing a quantitative assessment of the model’s performance. Finally, researchers and developers investigate the results of the evaluation and adjust parameters as needed to enhance the quality of the recommendation system. This iterative process of evaluation, adjustment, and refinement helps optimize the system for improved accuracy, relevance, and user satisfaction over time.

4.3 Training and Testing Splitting

When designing the training and testing split for evaluating a recommendation system, several considerations are crucial to ensure an effective assessment of system performance. Firstly, considering only users who have
taken more than one course ensures that there is sufficient data available for each user to conduct meaningful analysis and evaluation. This approach helps in capturing diverse user preferences and behaviours, enhancing the reliability of the evaluation results. Secondly, dividing the courses into two parts individually according to specified proportions is essential. This step ensures that each user’s data, including their interactions with courses, is appropriately divided based on the specified ratio. This division helps maintain a balance between training and testing data, preventing biases and ensuring that the model learns from a representative sample of user interactions. Finally, splitting the dataset corresponding to the curriculum based on the proportions defined in the previous step further enhances the evaluation process. This division ensures that the dataset is divided into two distinct parts: one for training the recommendation system and the other for testing its performance. By separating the training and testing datasets based on curriculum proportions, researchers and developers can conduct a comprehensive evaluation that reflects real-world usage scenarios and provides reliable insights into the system’s effectiveness in recommending courses to users.

4.3 Accuracy Management

By incorporating both hit rate and F1 score as performance measurements, researchers and developers gain a comprehensive understanding of the recommendation system’s effectiveness. The hit rate focuses on user interaction and satisfaction, while the F1 score provides insights into the system’s accuracy and completeness in making recommendations. Together, these metrics offer valuable insights for optimizing the recommendation system and improving the quality of course recommendations for users.

4.3.1 Hit Rate Calculation

The hit rate is a metric that measures the percentage of users who receive at least one relevant recommendation from the system. It reflects the system’s ability to provide personalized and valuable suggestions to users, impacting user engagement and satisfaction. A higher hit rate indicates better performance in delivering relevant content to users, enhancing the overall effectiveness of the recommendation system.

\[
\text{HitRate} = \frac{\text{Number of Hits}}{\text{Number of Users}}
\]

The hit rate of a recommendation system depends on the number of recommended courses provided. A larger number of recommendations can improve the hit rate by offering more opportunities to include relevant items. However, the quality and relevance of the recommendations are also essential factors that influence the hit rate. Simply increasing the number of recommendations may not guarantee a higher hit rate if the recommendations are not tailored to the user’s preferences and needs.

4.3.2 F1 Score Calculation

The F1 score is a crucial metric in evaluating the performance of a recommendation system, as it combines both precision and recall providing a balanced assessment. Precision measures the accuracy of the system’s positive predictions, specifically how many of the recommended courses are relevant to the user. Recall, on the other hand, evaluates the system’s ability to identify all relevant courses among those available.

\[
\text{Recall} = \frac{\text{Number of correctly predicted items}}{\text{Total number of relevant items}}
\]

\[
\text{Precision} = \frac{\text{Number of correctly predicted items}}{\text{Total number of recommended items}}
\]

\[
\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]

The F1 score is calculated using the harmonic mean of precision and recall, ensuring that both metrics contribute equally to the final score. This harmonic mean nature prevents one metric from dominating the
score, making the F1 score a reliable measure of the system’s overall effectiveness in providing accurate and comprehensive recommendations.

\[
F1 \text{ Score}_{\text{average}} = \frac{\sum_{i=1}^{n} F1 \text{ Score}_i}{\text{Number of Users}}
\]

A high F1 score is achieved when both precision and recall values are close to 1, indicating that the recommendation system is successfully balancing accuracy and completeness in its recommendations.

5. EXPERIMENTATION AND RESULTS

In the experimentation phase of our project, we evaluated the performance of our recommendation system in providing personalized course recommendations to users. Our machine learning application, hosted on PyPI under the name ‘isne-recommendation,’ comprises three key components: TfidfLinearKernel, FeatureRatingsKNN, and Hybrid classes. To maintain consistency and compatibility with our system, the dataset implemented in our project follows a specific pattern as an example table below.

**Table 1: A user dataset pattern**

<table>
<thead>
<tr>
<th>username</th>
<th>education</th>
<th>age</th>
<th>email</th>
<th>address</th>
<th>Course</th>
<th>date</th>
<th>payment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Julian Cook</td>
<td>Bachelor degree</td>
<td>20</td>
<td>cmu.ac.th</td>
<td>Canada</td>
<td>Calculus</td>
<td>2024-01-01 15:04:00</td>
<td>success</td>
</tr>
<tr>
<td>Nolan Jones</td>
<td>Middle school degree</td>
<td>27</td>
<td>gmail.com</td>
<td>Data Structure</td>
<td>2024-10-24 10:25:00</td>
<td>failure</td>
<td></td>
</tr>
<tr>
<td>Serenity Parker</td>
<td>High school degree</td>
<td>41</td>
<td>hotmail.com</td>
<td>California</td>
<td>English</td>
<td>2024-07-10 16:18:00</td>
<td>disapproval</td>
</tr>
</tbody>
</table>

**Table 2: An item dataset pattern**

<table>
<thead>
<tr>
<th>course</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculus</td>
<td>Introduction to vector, derivatives of functions of one variable and applications, indefinite and definite integrals.</td>
</tr>
<tr>
<td>English</td>
<td>Communication in English for everyday interactions. More advanced listening, speaking, reading and writing.</td>
</tr>
</tbody>
</table>

However, for illustrative purposes, we used example datasets available at 'https://startg2545.github.io/item_tutorial.csv' for item data and 'https://startg2545.github.io/user_item_tutorial.csv' for user data. During model evaluation, we emphasized the importance of splitting the dataset into training and testing sets. Commonly used ratios for this split include 60:40, 70:30, and 80:20. The training set is used to train the model to recognize patterns in course descriptions, while the testing set is used to evaluate the model’s performance without bias.

We evaluated the model’s performance using metrics such as hit rate and F1 score. The hit rates achieved were \textbf{14.69\%}, \textbf{48.24\%}, and \textbf{69.18\%} for the first approach, second approach, and hybrid approach, respectively. Similarly, the F1 scores were \textbf{2.58\%}, \textbf{7.92\%}, and \textbf{11.62\%} for the same approaches, indicating the accuracy of our recommendation system across different methodologies. By incorporating these evaluation metrics, we gained insights into the effectiveness of our recommendation system in providing personalized course recommendations, contributing to the overall success of our project.
6. EXPERIMENTATION AND RESULTS

In this project, I integrate insights from diverse studies to develop a recommendation system grounded in comprehensive research. The system is designed to recommend the most suitable courses for learners who have previously taken at least one course. It offers personalized recommendations, conducts thorough user data analysis to enhance accuracy, and contributes to improving the overall learning experience. However, it’s important to note that the system’s applicability is currently limited to online learning platforms that adhere to specific data formats required for implementation.

6.1 First Approach Analysis

The first approach has relatively low accuracy in both Hit Rate and F1 Score, indicating that the model’s performance in recommending relevant courses to users is not very effective. A Hit Rate of **14.69%** means that only about **14.69%** of the recommended courses match with the actual courses users take. Similarly, an F1 Score of **2.58%** suggests poor precision and recall, further highlighting the limitations of this approach in accurately predicting user preferences.

6.2 Second Approach Analysis

The second approach shows improved accuracy compared to the first approach, with a Hit Rate of **48.24%** and an F1 Score of **7.92%**. This indicates that the model in the second approach performs better in recommending courses that users are likely to take. However, the accuracy is still relatively moderate, suggesting room for further improvement.

6.3 Hybrid Approach Analysis

The hybrid approach demonstrates the highest accuracy among the three approaches, with a Hit Rate of **69.18%** and an F1 Score of **11.62%**. This indicates that combining multiple recommendation techniques, such as content-based filtering and collaborative filtering (as implied by the hybrid approach), leads to more accurate and personalized recommendations for users. The higher Hit Rate and F1 Score suggest that the hybrid approach has a better understanding of user preferences and is more successful in recommending relevant courses.

6.4 Challenges

The challenges encountered in the project are diverse, with the most significant being the process of gathering and comprehending studies related to the recommender system. We conducted a deep investigation into the evidence provided by authors to ensure reliability. Additionally, our main obstacle lies in Machine Learning Operations (MLOps) concerning deployment, as we are unfamiliar with the process.

6.5 Suggestions and further improvements

A suggestion for improving the project is to expand its compatibility with a broader range of online learning platforms. This would reduce the concerns developers have about data consistency between their datasets and the recommender system, making it more accessible and user-friendly across various platforms.

6.6 Essential Knowledge Applied in This Project

For developers who wish to comprehend data analysis, machine learning, and generative AI for understanding user item configurations in large datasets. They should further acquaint knowledge such as data sparsity [23], the cold start problem [24], scalability issues, algorithm complexity, proposing solutions like data augmentation, cold start handling techniques, scalability improvements, and algorithmic enhancements. These perceptions encompass user-item configuration, behavior analysis, and item characteristics, crucial for addressing challenges like data sparsity and scalability [25]. It ensures the development of robust recommendation systems that enhance user experiences across various domains.
7. ACKNOWLEDGEMENT

I (Mr. Newin Yamaguchi) would like to express my deepest gratitude to all those who have contributed to the completion of this project. Without their support, this endeavor would not have been possible. First and foremost, I would like to thank my advisor (Mr. Kampol Woradit) for their guidance and invaluable insights throughout the duration of this project. Their expertise and encouragement have been instrumental in shaping the direction of my research. Furthermore, I would like to acknowledge the support of my colleagues (Ms. Patcharaporn Satantaipop) for their constructive feedback and encouragement during the writing process. I am deeply thankful to my friends and family for their unwavering support and understanding throughout this journey. Their encouragement has been a constant source of motivation. Finally, I would like to express my gratitude to the Department of Computer Engineering, Chiang Mai University for their financial support, which made this project possible.

8. REFERENCES


