PROJECT No. ISNE P701-2/2567

Recommendation System

Newin Yamaguchi	**630615028**
Patcharaporn Satantaipop	**630615035**

A Project Submitted in Partial Fulfillment of Requirements
for the Degree of Bachelor of Engineering
Department of Computer Engineering
Faculty of Engineering
Chiang Mai University
2024

โครงงานเลขที่ วศ.สค. P701-2/2567

เรื่อง

ระบบช่วยแนะนำ

โดย

นายเนวิน ยามากุชิ รหัส 630615028
นางสาวพัชราภรณ์ สท้านไตรภพ รหัส 630615035

โครงงานนี้
เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่
ปีการศึกษา 2567

Project Title	: Recommendation System	
Name	: Newin Yamaguchi	630615028
	Patcharaporn Satantaipop	630615035
Department	: Computer Engineering	
Project Advisor	: Asst. Prof. Kampol Woradit, Ph.D.	
Degree	: Bachelor of Engineering	
Program	: Information Systems and Network Engineering	
Academic Year	: 2024	

The Department of Computer Engineering, Faculty of Engineering, Chiang Mai University has approved this project to be part of the degree of Bachelor of Engineering (Information Systems and Network Engineering)

. . . . Department chair

(Assoc. Prof. Santi Phithakkitnukoon, Ph.D.)

Project examination committee:

. . . . Main advisor / Chair

(Asst. Prof. Kampol Woradit, Ph.D.)

. . . . Committee member

(Asst. Prof. Yuthapong Somchit, Ph.D.)

. . . . Committee member

(Assoc. Prof. Kenneth Cosh, Ph.D.)

หัวข้อโครงงาน	: ระบบช่วยแนะนำ	
	: Recommendation System	
โดย	: นายเนวิน ยามากุชิ	รหัส 630615028
	นางสาวพัชราภรณ์ สท้านไตรภพ	รหัส 630615035
ภาควิชา	: วิศวกรรมคอมพิวเตอร์	
อาจารย์ที่ปรึกษา	: ผศ.ดร. กำพล วรดิษฐ์	
ปริญญา	: วิศวกรรมศาสตรบัณฑิต	
สาขา	: วิศวกรรมระบบสารสนเทศและเครือข่าย	
ปีการศึกษา	: 2567	

บทคัดย่อ

โครงงานนี้สำรวจวิธีการที่มีประสิทธิภาพและรวดเร็วในการนำเสนอหลักสูตรออนไลน์ แก่ผู้เรียนในปัจจุบัน การเรียนรู้ออนไลน์ได้รับความนิยมอย่างกว้างขวางเนื่องจากมีราคาไม่แพงและเข้าถึงได้ การจัดหาหลักสูตรที่ตอบสนองความต้องการของผู้เรียนได้ทันท่วงทีเป็นสิ่งสำคัญอย่างยิ่งในการป้องกันการรบกวนสมาธิจากการเรียน

ด้วยการบูรณาการแนวทางแก้ไขผลกระทบและข้อค้นพบจากการวิจัยที่เกี่ยวข้อง โครงการนี้เน้นย้ำถึงบทบาทที่สำคัญของระบบช่วยแนะนำในการดึงดูดผู้เรียนให้มุ่งเน้นไปที่เส้นทางการเรียนรู้และบรรลุเป้าหมาย โดยเน้นย้ำถึงการเข้าถึงทางเศรษฐกิจและโอกาสสำหรับบุคคลทั่วไปในการมีส่วนร่วมกับทักษะอาชีพที่จำเป็น ข้อกังวลเหล่านี้มีความสำคัญเป็นพิเศษเมื่อพิจารณาจากค่าใช้จ่ายที่สูงของการศึกษาแบบดั้งเดิมและการรับรู้ถึงความมีอภิสิทธิ์ของสถาบันบางแห่ง ซึ่งสร้างอุปสรรคสำหรับผู้เรียนที่ด้อยโอกาสทางเศรษฐกิจ

Project Title	: Recommendation System
Name	: Newin Yamaguchi 630615028
	Patcharaporn Satantaipop 630615035
Department	: Computer Engineering
Project Advisor	: Asst. Prof. Kampol Woradit, Ph.D.
Degree	: Bachelor of Engineering
Program	: Information Systems and Network Engineering
Academic Year	: 2024

ABSTRACT

This project explores efficient and rapid solutions for delivering online courses to learners. In today's digital age, online learning has gained widespread popularity due to its affordability and accessibility. Providing courses that cater to the needs of learners in a timely manner is crucial to prevent distractions from studies.

By integrating solutions, impacts, and findings from related research. 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.

Acknowledgments

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.

Newin Yamaguchi
Patcharaporn Satantaipop
24 February 2024

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Project rationale

Traditional recommendation system problems

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.

Many e-Learning platforms lack sophisticated algorithms for suggesting courses based on learners' preferences. Consequently, users frequently resort to selecting courses prominently displayed on the homepage, leading to a detrimental impact on study intention.

1.2 Objectives

1. To provide personalized course recommendations for learners.
2. To collect and analyze user data to improve course recommendations.

1.3 Project scope

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.

1.3.1 Approaches

Our recommendation system aims to enhance user experience by predicting ratings for specific items based on individual preferences. We will utilize content-based filtering, analyzing user features and preferences to recommend items, and collaborative filtering, leveraging past user interactions to personalize recommendations.

1.3.2 Libraries

- **Scikit-learn:** Utilized for predictive data analysis, providing a vast collection of machine learning algorithms.
- **SciPy:** Useful for solving mathematical equations and algorithms.
- **NumPy**: Fundamental for scientific computing in Python.
- **Pandas:** Providing fast and flexible data structures for data analysis.

1.3.3 Dataset

Kaggle

Kaggle will be used as a source of quality datasets for building AI models, allowing users to explore, analyze, and share data.

1.4 Expected outcomes

1. `Improved user experience and satisfaction.`
2. `Enhanced content discovery.`

1.4.1 Hardware technology

1. **Cloud Instance:** Required for running recommendation algorithms and serving recommended items.
2. **Graphics Processing Units (GPUs):** Useful for speeding up the training process for deep learning-based recommendation systems.
3. **Storage:** Necessary for storing user behavior data, item features, and trained models.
4. **Memory:** Crucial for storing intermediate computations and caching frequently accessed data.

1.4.2 Software technology

1. **Visual Studio Code:** Used for building recommendation systems with Python.
2. **Codespace:** Cloud-hosted development environment.
3. **Microsoft Excel:** Utilized for dataset processing.
4. **APIs:** For modular and scalable architectures.
5. **Deployment:** Microsoft Azure for packaging and managing components.

1.5 Project plan

Task	Jun 2023	Jul 2023	Aug 2023	Sep 2023	Oct 2023	Nov 2023	Dec 2023	Jan 2024	Feb 2024	Mar 2024
Research/Discuss	■	■	■	■						
Testing/Experiment			■	■	■					
Implement				■	■	■	■	■		
Draft Report									■	
Final Report										■

Table 1.1: Gantt chart

1.6 Roles and responsibilities

This project is made possible by 2 students and 1 adviser

- `Newin Yamaguchi`: Responsible for integration, scope, time, data structure, and collaboration.
- `Patcharaporn Satantaipop`: Responsible for forecasting, tools, datasets, and conclusion.
- `Kampol Woradit`: Adviser providing suggestions and support.

1.7 Impacts of this project on society, health, safety, legal, and cultural issues

The project aims to reduce decision-making time for users and enhance efficiency, ultimately benefiting society by meeting users' needs more effectively and improving the quality of products such as courses, movies, and videos.

Chapter 2
Background Knowledge and Theory

This chapter provides an overview of *e-Learning course recommendation systems*, focusing on the application of machine learning techniques. It draws upon relevant research articles to establish a comprehensive understanding of the topic, aiming to provide insight into the intricacies of these systems and their relevance in the field of machine learning.

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 behavior.

2.2 Content-Based Filtering

The first step is the *Feature Rating*, where features describe users and items in a recommendation system. 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)* are implemented to calculate the weights of courses' descriptions.

1. **Term Frequency (TF):**

$$\mathrm{tf}(\mathrm{term}, \mathrm{document}) = \frac{f(\mathrm{term}, \mathrm{document})}{\sum_{\mathrm{term}' \in \mathrm{document}} f(\mathrm{term}', \mathrm{document})} \tag{2.1}$$

2. **Inverse Document Frequency (IDF):**

$$\mathrm{idf}(\mathrm{term}, \mathrm{allDocuments}) = \log\left(\frac{N}{1 + \mathrm{df}(t)}\right) + 1 \tag{2.2}$$

3. **TF-IDF (Term Frequency-Inverse Document Frequency):**

$$\mathrm{tfidf}(\mathrm{term}, \mathrm{document}) = \mathrm{tf}(\mathrm{term}, \mathrm{document}) \times \mathrm{idf}(\mathrm{term}, \mathrm{allDocuments}) \tag{2.3}$$

2.2.1 Advantages

1. Quick processing as it focuses on individual users without considering others.
2. Capable of capturing specific interests of niche user groups.

2.2.2 Disadvantages

1. Results depend on feature definition and user knowledge.
2. Limited to the user's current interests, lacking the ability to expand interests.

2.3 Collaborative Filtering

Collaborative filtering recommends items based on the ratings of similar users. *K-Nearest Neighbors (KNN)* is employed for this purpose.

K-Nearest Neighbors (KNN):

- `Constructs a user-item matrix.`
- `Calculates cosine distances between courses.`
- `Selects top similar courses based on cosine distances.`

2.3.1 Advantages

1. Does not require feature definition.
2. Can recommend different items without specifying features.

2.3.2 Disadvantages

1. New items may not be recommended until users rate them.
2. Accuracy decreases in sparse matrices.
3. Tends to recommend popular courses over less attended ones.

2.4 Hybrid Filtering

One common thread in recommender systems is the need to combine recommendation techniques to achieve peak performance. All of the known recommendation techniques in different ways.

Technique	Background	Input	Process
Collaborative	Ratings from U of items in I.	Ratings from u of items in I.	Identify users in U similar to u, and extrapolate from their ratings of i.
Content-based	Features of items in I	u‘s ratings of items in I	Generate a classifier that fits u‘s rating behavior and use it on i.
Demographic	Demographic information about U and their ratings of items in I.	Demographic information about u	Identify users that are demographically similar to u, and extrapolate from their ratings of i.
Utility-based	Features of items in I.	A utility function over items in I that describes u‘s preferences.	Apply the function to the items and determine i‘s rank.
Knowledge-based	Features of items in I. Knowledge of how these items meet a user‘s needs.	A description of u‘s needs or interests.	Infer a match between i and u‘s need.

Table 2.1: Recommendation Techniques.

Hybrid recommendation systems combine multiple techniques to enhance performance by customizing recommendations based on specific conditions and dataset requirements.

2.5 ISNE knowledge used, applied, or integrated in this project

Various aspects of Computer Engineering knowledge are integrated into the project, including basic programming, object-oriented programming, data structures and algorithms, and fundamentals of database systems. Some of the key areas of knowledge applied include:

2.5.1 Basic Computer Programming for Information Systems and Network Engineering

C++ enables efficient algorithm implementation and performance optimization for recommendation systems. Basic programming skills aid in data preprocessing and custom component development. Additionally, C++ facilitates integration with existing systems and supports parallelization for improved computational efficiency.

2.5.2 Object-Oriented Programming

Object-Oriented Programming (OOP) allows for modular design, facilitating the organization of recommendation system components into reusable and understandable classes. Encapsulation ensures data integrity and abstraction simplifies complex algorithms, enhancing maintainability and scalability.

2.5.3 Data Structures and Algorithms

Data Structures optimize storage and retrieval of user-item interactions, enhancing recommendation efficiency. Algorithms like collaborative filtering or matrix factorization enable personalized recommendations based on user preferences. Efficient implementation of sorting and searching algorithms enhances recommendation performance, ensuring timely and relevant suggestions.

2.5.4 Fundamentals of Database Systems

Understanding database fundamentals aids in designing efficient storage schemas for user preferences, item attributes, and interaction data. Proficiency in database management ensures robust data retrieval and manipulation, supporting accurate recommendation generation. Knowledge of transaction management and concurrency control enhances data integrity and system reliability in handling user interactions.

2.6 Extracurricular knowledge used, applied, or integrated in this project

Understanding Python programming is crucial for our project since we'll be primarily using it to develop the recommendation system. Additionally, familiarizing ourselves with various libraries is essential, as they are key tools for implementing different functionalities within the system. Selecting and utilizing the right libraries will be pivotal in ensuring the effectiveness and efficiency of our system.

To achieve this, it is vital to have in-depth knowledge of data analysis, machine learning, and generative AI. These areas will help us understand the configuration, behavior, and characteristics of user items from large datasets. Thus, enabling us to provide personalized recommendations by learning from the data.

Chapter 3
Project Structure and Methodology

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 recommendations into a web application for user interaction and visualization of results. Here's a step-by-step summary of the experimentation and results:

3.1 Data Cleaning

1. Load user and item datasets that are compatible with the specified format.
2. In the user dataset, reformat dates, standardize payment statuses and education levels, clean addresses, and extract email domains. Moreover, rename all columns
3. In the item dataset, rename course and description columns
4. After finishing all processes, save it to new files.

name	**degree**	**Old**	**your mail**	**residence**	**course**	**enrollment**	**Is it paid**
John	4-year	26	john@gmail.com	Bangkok	OOP	yesterday	success
Peter	PHD	30	peter@cmu.ac.th		Web	recent days	failure
thomas	mid	42	nelson@msn.com	Chiang Mai	OS	a minute	disapprove

Table 3.1: Before cleaning the user dataset

username	**education**	**age**	**email**	**address**	**course**	**time**	**payment**
John	Bachelor degree	26	gmail.com	Bangkok	OOP	1/1/2566	success
Peter	Master degree	30	cmu.ac.th		Web App	1/5/2566	failure
thomas	High school level	42	msn.com	Chiang Mai	OS	2/2/2566	disapprove

Table 3.2: After cleaning the user dataset

Our Course Name	The Explanation
OOP	Object-Oriented Programming (OOP) allows for modular design, facilitating the organization of recommendation system components into reusable and understandable classes. Encapsulation ensures data integrity and abstraction simplifies complex algorithms, enhancing maintainability and scalability.
Web App	A web application is a software program accessed via web browsers over the internet. It offers various services, from basic websites to complex systems. It uses a client-server architecture for interaction, allowing modular design for scalability and flexibility. Technologies like HTML, CSS, JavaScript, and server-side scripting languages enable development.
OS	An operating system (OS) is software managing computer resources and providing a user interface. It handles tasks such as process, memory, file, and device management, enabling multitasking and efficient resource allocation. Encapsulation ensures data integrity, simplifies hardware interactions, and enhances maintainability and scalability. Examples include Windows, macOS, Linux, and Unix.

Table 3.3: Before cleaning the item dataset

Course	Description
OOP	Object-Oriented Programming (OOP) allows for modular design, facilitating the organization of recommendation system components into reusable and understandable classes. Encapsulation ensures data integrity and abstraction simplifies complex algorithms, enhancing maintainability and scalability.
Web App	A web application is a software program accessed via web browsers over the internet. It offers various services, from basic websites to complex systems. It uses a client-server architecture for interaction, allowing modular design for scalability and flexibility. Technologies like HTML, CSS, JavaScript, and server-side scripting languages enable development.
OS	An operating system (OS) is software managing computer resources and providing a user interface. It handles tasks such as process, memory, file, and device management, enabling multitasking and efficient resource allocation. Encapsulation ensures data integrity, simplifies hardware interactions, and enhances maintainability and scalability. Examples include Windows, macOS, Linux, and Unix.

Table 3.4: After cleaning the item dataset

3.2 Course Recommendation using TF-IDF and Linear Kernel

1. Load the cleaned dataset.
2. Match courses between user dataset and item dataset to return descriptions.
3. Use TF-IDF vectorization calculate the weights of words in the course descriptions.
4. Apply linear kernel to calculate the cosine similarities between all courses.
5. Return the top N recommended courses for the given courses taken by a specific user.

3.2.1 Term Frequency

$$logNormalization(\text{term, document}) = 1 + log(f(term, document)) \quad (3.1)$$

Let's take an example of a course with the following description:

- **Doc**: `The programming language is difficult, The local language is easy.`

logTF("The, Doc)	->	1 + log(2) ≈ 1.3
logTF("programming, Doc)	->	1 + log(1) = 1
logTF("language, Doc)	->	1 + log(2) ≈ 1.3
logTF("is, Doc)	->	1 + log(2) ≈ 1.3
logTF("difficult, Doc)	->	1 + log(1) = 1
logTF("local, Doc)	->	1 + log(1) = 1
logTF("easy, Doc)	->	1 + log(1) = 1

3.2.2 Inverse Document Frequency

$$idf(term, allDocuments) = log\left(\frac{N}{df(term)}\right) \quad (3.2)$$

Let's take an example of 2 courses with the following description:

- **Doc1**: `The web programming language is taught by a foreign professor.`
- **Doc2**: `The network security is taught by a thai professor.`

idf("The", Doc1)	->	log(2/2) = 0
idf("web", Doc1)	->	log(2/1) ≈ 0.3
idf("network", Doc2)	->	log(2/1) ≈ 0.3
idf("professor", Doc2)	->	log(2/2) = 0

3.2.3 Term Frequency and Inverse Document Frequency

$$tfidf(term, document, allDocuments) = tf(term, document) \times idf(term, allDocuments) \quad (3.3)$$

Let's take an example of a word 'language' that appears in 2 documents out of 3 documents.

- **Doc1**: `The programming language is difficult, The local language is easy.`
- **Doc2**: `The web programming language is taught by a foreign professor.`
- **Doc3**: `The network security is taught by a thai professor.`

1. **tf**("language", Doc1) = 1 + log(2) ≈ 1.3
2. **idf**("language", [Doc1, Doc2, Doc3]) = log(3/2) ≈ 0.18
3. **tfidf**("language", Doc1, [Doc1, Doc2, Doc3]) = 1.3 × 0.18 ≈ 0.23

3.2.4 Cosine Similarity

$$cosine\_similarity(A, B) = \frac{A \cdot B}{\|A\| \cdot \|B\|} \quad (3.4)$$

Let's take an example of 3 courses with the following matrix where each row represents a course and each column represents a word:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
0	0.0	0.3	0.2	0.0	0.4	0.5	0.0	0.0	0.0	0.3	0.0	0.0	0.0	0.4	0.0
1	0.3	0.0	0.0	0.4	0.2	0.3	0.0	0.0	0.3	0.3	0.0	0.3	0.0	0.2	0.4
2	0.3	0.0	0.0	0.0	0.2	0.3	0.4	0.4	0.3	0.0	0.4	0.3	0.4	0.2	0.0

Table 3.5: User-Item Matrix

Compute the *dot product* between each pair of course vectors, so we get a course-course similarity matrix.

$$\begin{bmatrix} 1 & 0.4493628 & 0.20315676 \\ 0.4493628 & 1 & 0.44147846 \\ 0.20315676 & 0.44147846 & 1 \end{bmatrix}$$

Table 3.6: Item-Item Matrix

We can finally use the this matrix to recommend courses based on a given course.

3.3 Course Recommendation using Feature Ratings and KNN

1. Load the cleaned dataset.
2. Determine and calculate the feature ratings for each course given by users.
3. Train the model on the user-course to learn the relationships between courses based on user preferences.
4. Predict and rank the distances of courses based on the courses that the user has taken.
5. Return the top N recommended courses for the given user.

3.3.1 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 behaviors of users in each column as follows.

- **`Email`** `Column`
  - `Give 0 points if no email is provided.`
  - `Give 1 point if a common email domain is provided.`
  - `Give 2 points if an educational email domain is provided.`
- **`Age-Education`** `Column`
  - `Give 0 points if a person is in the educational system.`
  - `Give 1 point if a person just graduated.`
  - `Give 2 points if a person is in the working age.`
- **`Time`** `Column`
  - `Give 0 points if the time registration is unsuitable for a learning period.`
  - `Give 1 point if the time registration is suitable for a learning period.`
- **`Payment`** `Column`
  - `Give 0 points if the payment is overdue.`
  - `Give 1 point if the payment is unapproved.`
  - `Give 2 points if the payment is on time.`

- `Address` Column
  - Give 0 points if the address is blank.
  - Give 1 point if the address is filled.

Find the *impressive level* individually by considering the historical background of the user and the course.

User	Course	Email	Age-Education	Time	Payment	Address	Score
User 1	Course 1	0	0	1	1	1	2.75
User 2	Course 1	1	1	1	2	0	3.75
User 3	Course 3	2	0	0	0	0	1.75
User 4	Course 3	0	1	0	2	1	3.5
User 5	Course 2	1	2	0	0	0	1.875

Table 3.7: Feature Ratings

Remove the feature rating columns to remain only user, course, and score.

User	Course	Score
User 1	Course 1	2.75
User 2	Course 1	3.75
User 3	Course 3	1.75
User 4	Course 3	3.5
User 5	Course 2	1.875

Table 3.8: Cleaned Feature Ratings

Pivot the table to get the user-course matrix.

	User 1	User 2	User 3	User 4	User 5
Course 1	2.75	3.75	NaN	NaN	NaN
Course 2	NaN	NaN	NaN	NaN	1.875
Course 3	NaN	NaN	1.75	3.5	NaN

Table 3.9: User-Course Matrix

3.3.2 Nearest Neighbors Model

Fit a k-nearest neighbors (KNN) model. Where yellow, red, and blue colours represent 3 different courses.

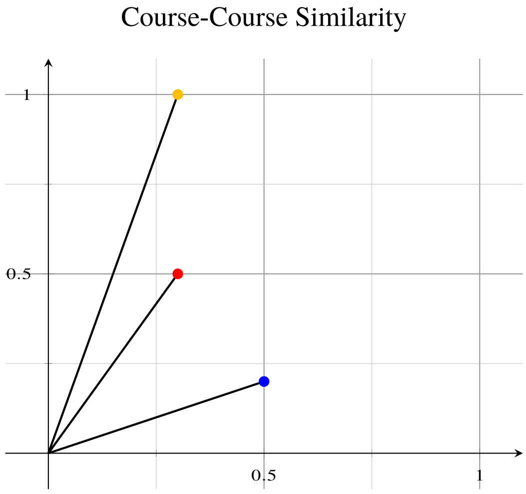

Figure 3.1: KNN Model

Use the KNN model to calculate the cosine distance between the red course and all other courses.

Course	Cosine Distance
yellow	0.25
blue	0.4

Table 3.10: Predicted Distances

After calculating the cosine distances for all courses, select the top most similar courses.

3.4 Hybrid Course Recommendation

1. Loaded the cleaned dataset.
2. Combined the results from TF-IDF and KNN approaches using a hybrid recommendation system.
3. Returned the top N recommended courses for the given user, integrating recommendations from both TF-IDF and KNN approaches.

3.4.1 The combination of TF-IDF and KNN

Set the weights and normalize the matrices of TF-IDF and KNN.

	User 1	**User 2**	**User 3**	**User 4**	**User 5**
Course 1	1.00	3.75	2.00	2.50	3.25
Course 2	2.25	2.75	1.75	1.50	1.87
Course 3	1.00	1.40	1.95	4.55	4.45

Table 3.11: Before the normalization

	User 1	**User 2**	**User 3**	**User 4**	**User 5**
Course 1	0.167	0.626	0.334	0.417	0.543
Course 2	0.486	0.594	0.378	0.324	0.404
Course 3	0.145	0.204	0.284	0.662	0.647

Table 3.12: After the normalization

Stack arrays in sequence horizontally

	Column 1	Column 2
Row 1	0.1	0.2
Row 2	0.3	0.4
Row 3	0.5	0.6

Table 3.13: Matrix 1

	Column 3
Row 1	0.7
Row 2	0.8
Row 3	0.9

Table 3.14: Matrix 2

	Column 1	Column 2	Column 3
Row 1	0.1	0.2	0.7
Row 2	0.3	0.4	0.8
Row 3	0.5	0.6	0.9

Table 3.15: Stacked Matrix

3.4.2 The process of Recommendation

We apply Nearest Neighbors to the stacked matrix to get the final recommendation. Nevertheless, the process of recommendation is the same as the KNN approach.

3.5 Evaluation Method

1. Consider users who have taken more than one course.
2. The training and testing datasets are splitted by identifying the ratio.
3. Allow the model to familiarize itself with the training dataset to ensure there is no bias.

4. Predict the courses for all users using the trained model.
5. Calculate the accuracy by measuring the similarities between the predicted courses and the actual test courses.
6. Investigate the results and adjust the parameters to enhance the quality of the system.

3.5.1 Training and Testing

This method is designed to split a dataset of user-course interactions into training and testing sets, ensuring that each courses of users are appropriately divided for machine learning model training and evaluation.

1. **Consider only users who have taken ore than 1 course:** This step ensures that you have enough data from each user for meaningful analysis.
2. **Divide the courses into 2 parts individually according to the specified proportions:** This ensures that each user's data is divided according to the specified proportion.
3. **Split the dataset corresponding to the curriculum from step 2:** This ensures that the dataset is approximately divided into two parts: one for training and one for testing.

3.5.2 Accuracy Measurement

There are two performance measurements in total including hit rate and f1 score.

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 actually 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. Hit Rate is calculated as follows:

$$\text{HitRate} = \frac{Numberofhitusers}{Allusers} \tag{3.5}$$

A larger value of recommended courses can potentially improve the hit rate because it provides more opportunities to include relevant items. Therefore, the performance of the hit rate indeed depends on the number of recommended courses.

F1 Score

This is a metric used to evaluate the performance of a classification model. It measures the proportion of true positive predictions among all positive predictions made by the model and measures the proportion of true positive predictions among all actual positive cases in the data simultaneously. To find the F1 Score from Recall and Precision, Recall, Precision, and F1 Score can use the following formulas:

$$\text{Recall} = \frac{Number of correctly predicted items}{Total number of relevant items} \tag{3.6}$$

$$\text{Precision} = \frac{Number of correctly predicted items}{Total number of recommended items} \tag{3.7}$$

$$\text{F1} = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{3.8}$$

- Number of correctly predicted items - The course that system predict correctly.
- Total number of relevant items - The courses that have been taken by the user from the course recommendation system
- Total number of recommended items - All courses that the recommendation system suggests for user

To find the average F1 score from all users by

$$F1_{average} = \frac{\sum_{i=1}^{n} F1_i}{Number of Users} \tag{3.9}$$

The harmonic mean nature makes sure if either Precision or Recall has a really high value, then it does not dominate the score. F1 Score has a high value when both precision and recall values are close to 1.

Chapter 4
Experimentation and Results

In this section, we delve into the experimentation phase of our project, where the proposed methodologies were put to the test, and their efficacy was evaluated. The primary objective was to assess the performance and suitability of the recommendation system in providing personalized course recommendations to users.

4.1 Upload dataset

The dataset provided by the developer is required to be a csv format and consists of two main components:

1. **User Dataset:** This includes the user's profile, which mostly contains 9 information that are username, course, email, age, education, time, payment, address, and score they have taken in the past. The user's profile is used to generate personalized recommendations.
2. **Course Dataset:** This mainly includes 2 information about the courses available on the platform that are course name and description. This data is used to generate course recommendations.

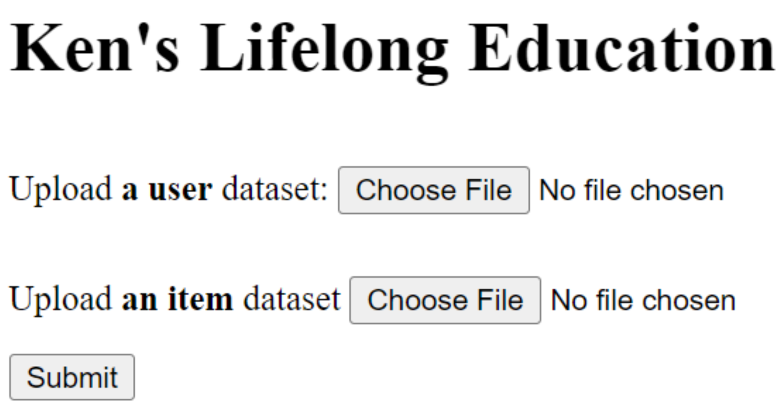

Figure 4.1: Uploading dataset

4.2 Input Data

The input data of the recommendation system is a name of user who is considered to be recommended. This person is typically a user who has taken at least one course on the platform. The recommender is applied by 3 different techniques including of 1) TF-IDF and Linear Kernel approach 2) Feature Ratings and KNN approach, 3) the combination of 2 previous approaches.

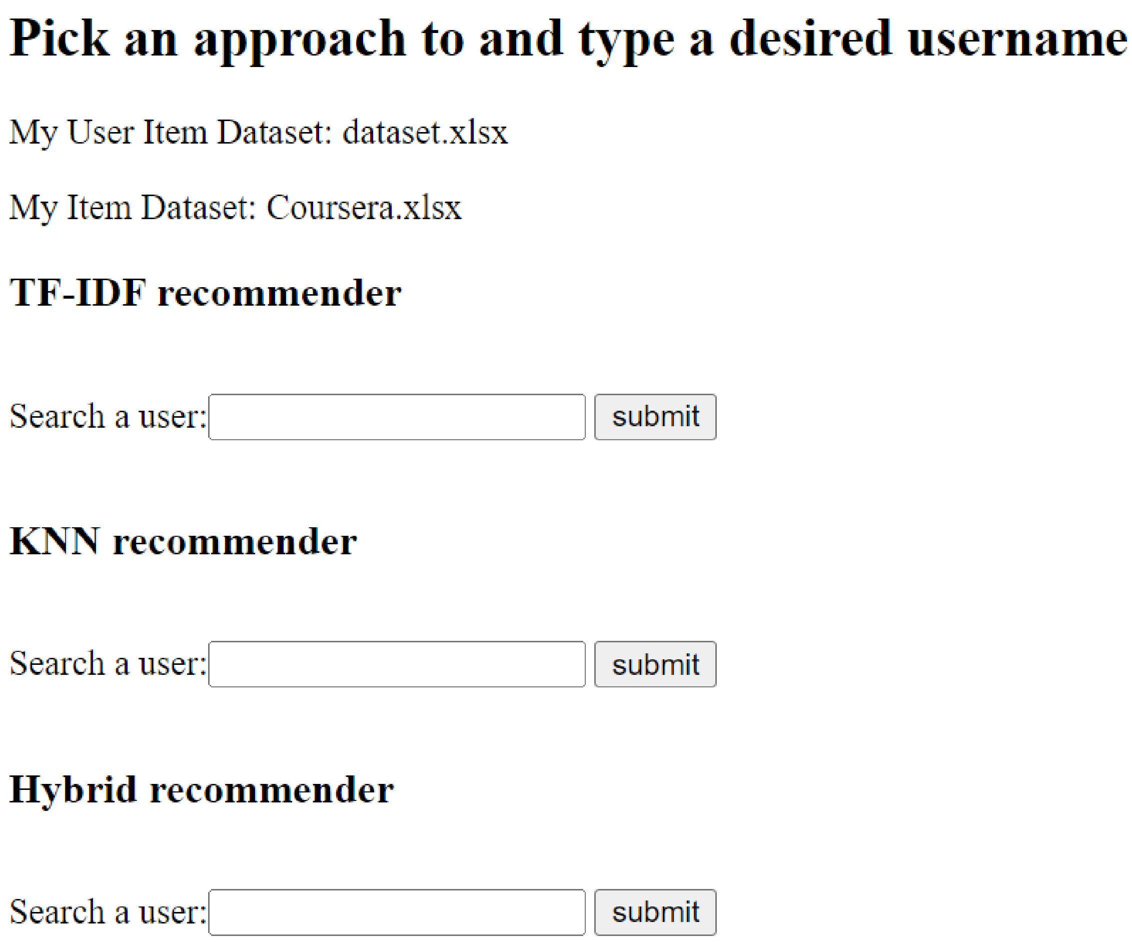

Figure 4.2: Input username

4.3 Output Data

The output data of the recommendation system is a list of recommended courses for a given user. The list is sorted by the score of the suggestions, which is calculated by one of the 3 techniques. The score is used to determine the relevance of the course to the user's profile.

Recommended for you

Model: tfidf

User: Martha Long

Taken courses:

- Build Basic Generative Adversarial Networks (GANs)
- Non-Equilibrium Applications of Statistical Thermodynamics
- Emergency Care: Pregnancy, Infants, and Children

Recommendation:

Course	Score
Addiction Treatment: Clinical Skills for Healthcare Providers	0.261621
Write A Feature Length Screenplay For Film Or Television	0.226188
Physics of silicon solar cells	0.210851
Internet of Things: How did we get here?	0.205599
Python and Machine Learning for Asset Management	0.190557
Medical Applications of Particle Accelerators (NPAP MOOC)	0.185854
Foundations of Public Health Practice: Behaviour & Behaviour Change	0.175814
Biomedical Visualisation	0.170230
COVID-19 - A clinical update	0.168129
Health Care IT: Challenges and Opportunities	0.166725

Figure 4.3: Output recommended courses

4.4 evaluation outcomes

We have selected the dataset to be used as follows

```python
import pandas as pd
i_data = pd.read_csv('https://startg2545.github.io/item_tutorial.csv')
ui_data = pd.read_csv('https://startg2545.github.io/user_item_tutorial.csv')
```

Figure 4.4: Dataset to evaluation

First approach

This approach is an entire content-based filtering technique that leverages the Term Frequency and Inverse Document Frequency to statistically measure the importance of words in the descriptions.

The model achieved a Hit Rate accuracy of 14.69%, signifying its effectiveness in predicting relevant items. Regarding the F1 Score, which balances precision and recall, the model attained an accuracy of 2.58%, highlighting its overall performance in recommendation accuracy and error handling.

Second Approach

This approach involves using Feature Ratings and applying them to the K-Nearest Neighbors (KNN) algorithm in a recommendation system. However, this is a combination of content-based and collaborative filtering techniques.

The model attained a Hit Rate accuracy of 48.24%, demonstrating its capability to predict relevant items. In terms of the F1 Score, which considers precision and recall, the model achieved an accuracy of 7.92%, indicating its overall performance in recommendation accuracy and error management.

Hybrid Approach

The third approach combines TF-IDF content analysis with feature-based KNN, resulting in accurate and personalized recommendations for users.

The model achieved a Hit Rate accuracy of 69.18%, indicating its effectiveness in predicting relevant items. The F1 Score, which balances precision and recall, stood at 11.62%, reflecting its overall performance in recommendation accuracy and error minimization.

Chapter 5
Conclusions and Discussions

5.1 Conclusions

นศ. ควรสรุปถึงข้อจำกัดของระบบในด้านต่างๆ ที่ระบบมีในเนื้อหาส่วนนี้ด้วย

5.2 Challenges

ในการทำโครงงานนี้ พบว่าเกิดปัญหาหลักๆ ดังนี้

5.3 Suggestions and further improvements

ข้อเสนอแนะเพื่อพัฒนาโครงงานนี้ต่อไป มีดังนี้

Appendix A
The first appendix

This appendix provides additional technical details and information related to the implementation of the recommendation system aimed at facilitating course recommendations tailored to individual user interests on e-Learning platforms.

A.1 Data Collection and Preprocessing

We collected user interaction data from the e-Learning platform, including user ratings, course enrollments, and browsing history. The data were preprocessed to remove duplicates, handle missing values, and ensure consistency.

A.2 Content-Based Filtering

In the content-based filtering approach, we utilized the TF-IDF (Term Frequency-Inverse Document Frequency) technique to extract relevant features from course descriptions and user preferences. We then applied feature weighting and cosine similarity to recommend courses based on similarity to the user's preferences.

A.3 Collaborative Filtering

For collaborative filtering, we employed the K-Nearest Neighbors (KNN) algorithm to identify similar users or courses based on their interaction patterns. We calculated similarities between users or courses and generated recommendations by considering the preferences of similar users.

A.4 Hybrid Approach

The hybrid approach combines content-based and collaborative filtering techniques to leverage the strengths of both methods. We integrated the recommendations generated from both approaches using a weighted or ensemble method to provide more personalized and accurate course suggestions.

A.5 Model Evaluation

We evaluated the performance of the recommendation system using standard metrics such as accuracy, precision, recall, and F1-score. Additionally, we conducted A/B testing or user studies to assess user satisfaction and the effectiveness of the recommendations in improving the user experience and content discovery on the e-Learning platform.

A.6 Implementation Code Snippets

Below are excerpts from the implementation code:

```
# Sample code snippet for content-based recommendation
def content_based_recommendation(user_preferences, course_features, similarity_matrix, top_n=5):
    # Compute user profile based on preferences
    user_profile = compute_user_profile(user_preferences, course_features)

    # Calculate similarity between user profile and courses
    similarity_scores = similarity_matrix.dot(user_profile)

    # Get top N recommended courses
    top_courses_indices = np.argsort(similarity_scores)[::-1][:top_n]
    top_courses = [course_names[i] for i in top_courses_indices]

    return top_courses
```

```
# Sample code snippet for collaborative filtering recommendation
def collaborative_filtering_recommendation(user_id, user_course_matrix, similarity_matrix, course_names,
    # Get user vector from user-course matrix
    user_vector = user_course_matrix[user_id]

    # Calculate similarity between user and other users/courses
    similarity_scores = similarity_matrix.dot(user_vector)

    # Get top N recommended courses
    top_courses_indices = np.argsort(similarity_scores)[::-1][:top_n]
    top_courses = [course_names[i] for i in top_courses_indices]

    return top_courses
```

Appendix B
Manual

This appendix provides detailed information about the data collection and preprocessing steps conducted as part of the recommendation system project.

B.1 Data Sources

We collected data from multiple sources to build our recommendation system, including e-Learning platform logs, course metadata, and user profiles.

B.2 Data Preprocessing

The raw data underwent several preprocessing steps to ensure quality and consistency such as removal of duplicates, handling missing values, data cleaning, and data normalization.

B.3 Feature Engineering

We engineered additional features from the raw data to enhance the recommendation system's performance which consist of user-item interaction matrix, content features, and user profiles.

B.4 Data Exploration

Exploratory data analysis (EDA) was performed to gain insights into the dataset's characteristics including distribution of user ratings, course popularity, and user behavior patterns.

B.5 Data Integration

Finally, the preprocessed data were integrated into a unified dataset for model training and evaluation. The integrated dataset included user profiles, item features, and user-item interaction data necessary for building and testing the recommendation system.

Biographical Sketch

Name: Newin Yamaguchi
Affiliation: Chiang Mai University

Name: Patcharaporn Satantaipop
Affiliation: Chiang Mai University

Newin and Patcharaporn are students at the Department of Computer Engineering, Chiang Mai University, specializing in machine learning and algorithms. We are currently in the fourth-year of Bachelor's Degree. With a passion for data science, we have dedicated our career to exploring innovative solutions to motivate students to become more addicted to self-learning.

Our current project focuses on the development of a Recommendation System for e-Learning platforms. Recognizing the challenges associated with online learning. We aim to investigate diverse models to facilitate course recommendations tailored to individual user interests. By leveraging a hybrid approach that combines content-based and collaborative filtering techniques, we aim to enhance the content discovery process and improve user satisfaction in online education.

Overall, Newin and Patcharaporn are dedicated and driven researchers committed to making meaningful contributions to the field of Computer Engineering. Our recommendation system project for e-Learning platforms represents an ongoing commitment to improving online education and enhancing the learning experience for students worldwide.