Kingdom of Saudi Arabia
Ministry of Education
University of Jeddah
College of Computer Science and Engineering
Department of Computer Science and
Artificial Intelligence

المملكة العربية السعودية
وزارة التعليم
جامعة جدّة
كلية علوم وهندسة الحاسب
قسم علوم الحاسب والذكاء الاصطناعي

Course Project

CCAI 422 Recommender Systems

Phases 3

Group 3

Section R1

First Semester 2024/2025

Title: Content Based Filtering Recommendation System for DSA Questions

Team member:

Name	ID Number	Task
Luluwa Modeer	2111397	Code, Presentation and Report
Areej Showail	2111597	Code, Presentation and Report
Zahrah Rasheed	2113400	Code, Presentation and Report

1. Abstract

Mastering Data Structures and Algorithms (DSA) is a pivotal skill for students and professionals in computer science. However, the abundance of DSA problems often makes it difficult for learners to identify the right challenges that match their skill level and learning objectives. This project introduces a personalized recommendation system for DSA problems, leveraging content-based filtering (CBF) to suggest tailored problems based on attributes such as difficulty level, topic, and metadata. Initially using a curated dataset of 260 problems sourced from the GeeksForGeeks platform, we expanded the dataset by incorporating additional data, reaching a total of 740 problems. This expanded dataset allows the system to dynamically align its recommendations with the learner's proficiency and preferences.

The system's evaluation demonstrated encouraging results, achieving an overall accuracy of 87% and a weighted average f1-score of 0.86. Despite its success, the system showed varying performance across difficulty levels, excelling in "Medium" problems (f1-score: 0.93) but underperforming in "Basic" problems due to data imbalance. Hyperparameter tuning using Ridge regression improved metrics but led to overfitting, highlighting the need for further optimization.

To enhance the system's robustness and address limitations, this project implemented collaborative filtering and a hybrid model that combines content-based and collaborative filtering approaches. Notably, the development of these models was identified as future work in the paper; however, we incorporated them as an additional effort to enrich the recommendation system. These techniques integrate user interaction patterns with problem features to provide a more personalized learning experience. This project serves as a foundational step toward adaptive DSA learning systems, with future work focusing on refining hybrid models and incorporating real-world user feedback to enhance their effectiveness.

2. Introduction

In the ever-evolving domain of computer science, mastering Data Structures and Algorithms (DSA) is a cornerstone for students and professionals alike. However, navigating through the vast array of DSA topics and problems can be overwhelming, especially without a clear understanding of which problems align with one's skill level and learning objectives. Our project addresses this challenge by developing a personalized recommendation system for DSA questions, designed to guide learners in their journey to mastering these fundamental concepts.

The core of our recommendation system is content-based filtering (CBF), a method that evaluates the attributes of each problem, such as difficulty level, topic, and metadata. By analyzing these features, the system suggests tailored problems that match the learner's preferences and proficiency. Initially, the dataset comprised 260 problems sourced from the GeeksForGeeks platform via Kaggle. To enhance the system's effectiveness, the dataset was further expanded by collecting additional data, resulting in a more comprehensive collection of 740 problems.

The primary goal of this project is to provide a recommendation system that dynamically suggests problems based on features such as question difficulty level and the user's proficiency. This system is intended to empower learners, enabling them to focus on relevant problems that enhance their understanding and mastery of DSA concepts.

3. Related Work

Recommendation systems have become an essential part of modern technology, playing a significant role in various domains such as education, e-commerce, and entertainment. These systems enhance user experience by delivering personalized content and reducing information overload. In the educational context, particularly in learning Data Structures and Algorithms (DSA), recommendation systems help guide learners to relevant problems, enabling them to practice efficiently based on their skill levels and progress [1].

Content-Based Filtering (CBF) is one of the most popular approaches for developing recommendation systems. It evaluates item attributes—such as difficulty, topic, and metadata—to recommend items that align with user preferences. For instance, in this project, the CBF approach is employed to suggest DSA problems by analyzing features like question tags, difficulty levels, and topics. This method ensures personalized recommendations, as it focuses on the user's past interactions to recommend similar problems [2].

The dataset used in this project was sourced from Kaggle and contains approximately 260 problems from GeeksForGeeks. The dataset includes detailed attributes such as problem statements, difficulty levels, and associated topics, making it ideal for implementing an effective CBF-based system. While CBF excels in personalization, it faces challenges such as the cold start problem (insufficient data for new users or items), which can be mitigated by future integration of hybrid approaches combining CBF and Collaborative Filtering (CF) [3].

This project aims to develop a tailored recommendation system that empowers learners to navigate DSA topics effectively, setting the groundwork for further enhancements to improve recommendation accuracy and diversity.

4. Approach

Based on Content-Based Filtering (CBF), the suggested recommendation method analyzes and ranks DSA problems by utilizing Cosine Similarity and TF-IDF (Term Frequency-Inverse Document Frequency). This approach evaluates the problem features, such as difficulty level, topic, and metadata, to recommend questions that align with the learner's proficiency and preferences. The technique, the justification for choosing this strategy, the hyperparameters employed, and the assessment measures are all covered in depth below.

As an additional enhancement to the system, we also implemented **collaborative filtering** and a **hybrid recommendation model** that integrates collaborative filtering with content-based filtering. These extensions leverage both user interaction patterns and problem-specific features to improve recommendation quality and provide a more personalized learning experience.

4.1 Algorithm:

Our approach to developing the personalized recommendation system for Data Structures and Algorithms (DSA) questions revolves around the **Content-Based Filtering (CBF)** methodology. This method leverages the attributes of each problem, such as difficulty level to tailor recommendations that align with a user's proficiency and preferences. The following steps outline the process and the rationale behind key decisions

Steps Taken:

1. **Feature Extraction Using TF-IDF:**

- TF-IDF (Term Frequency-Inverse Document Frequency) is employed to assess the importance of terms in problem descriptions. Higher TF-IDF scores indicate that a term is more significant in characterizing the problem, allowing for more precise feature extraction.

2. **Similarity Computation:**

- Cosine Similarity is used to measure the similarity between the TF-IDF vectors of the user's preference profile and the available challenges. The similarity scores range from -1 to 1, with higher scores signifying closer matches between the user's interests and the problems.

$$\text{Cosine Similarity} = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\|\|\vec{B}\|}$$

3. **Recommendation Generation:**

- Problems with the highest similarity scores are recommended to the user, ensuring that the suggestions align closely with their proficiency and preferences.

4.2 Algorithm Selection

1. **Interpretation Ease:** The content-based approach makes it easy to understand why a certain issue was suggested, which is advantageous in educational settings.
2. **Low Cold-Start Dependency:** CBF doesn't need a lot of user data to make good recommendations, in contrast to collaborative filtering.
3. **Scalability:** For the specified dataset size (740 issues), TF-IDF and cosine similarity are computationally efficient.
4. **Customizability:** New features, such as trouble tags or user performance data, can be readily incorporated into the system.

4.3 Hyperparameters

1. TF-IDF Vectorizer Parameters:
    - **ngram_range= (1, 2)**: This hyperparameter specifies the range of n-grams (1-grams and 2-grams in this case) to be considered when creating features. By including both single words and pairs of consecutive words, it captures more context from the problem descriptions. This enhances the model's ability to understand and differentiate between problems.
2. Ridge Regression and Hyperparameter Tuning:
    - **alpha in Ridge Regression:** This is the regularization parameter. It controls the trade-off between minimizing the loss function (error) and the complexity of the model (overfitting). A smaller **alpha** allows the model to fit the data more closely, while a larger **alpha** adds more regularization, penalizing large coefficients to prevent overfitting.
    - **n_iter=20**: Indicates the number of random samples to be drawn from the specified parameter distribution. This allows efficient exploration of the parameter space without exhaustively testing every value.
    - **cv=5:** Specifies 5-fold cross-validation to evaluate the performance of the model for each set of hyperparameters. This ensures that the results are robust and not dependent on a specific train-test split.
    - **scoring='r2'**: Uses the R^2 score as the metric to evaluate model performance during hyperparameter tuning. A higher R^2 indicates a better fit to the data.

4.4 Evaluation Metrics

1. Accuracy
    - Calculate the percentage of problems that were accurately suggested out of all the recommendations that were provided.

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

2. Precision
    - Shows the percentage of pertinent suggested problems.

$$Precision\ (P) = \frac{TP}{TP + FP}$$

3. Recall
    - Captures the system's capacity to recognize all pertinent issues.

$$Recall\ (R) = \frac{TP}{TP + FN}$$

4. F1-score
    - Combines precision and recall giving a balanced evaluation.

$$F - Score\ (F) = \frac{2.P.R}{P + R}$$

4.6 Extra Work That Was Included In the Future Work of The Original Paper

Implementing content-based filtering (CBF), our approach incorporates collaborative filtering (CF) and a hybrid recommendation model that combines both methods. Collaborative filtering leverages user-item interaction data to recommend questions based on user preferences and shared similarities between items. Since our dataset did not include user interaction data, we generated dummy values to simulate user-item interactions. These dummy values allowed us to develop and evaluate the collaborative filtering component effectively, despite the absence of real interaction data.

- **Simulated User-Item Interaction Matrix**: We created a matrix with random ratings (dummy values) for simulated users, representing their interactions with various DSA questions.
- **Normalization of Ratings**: To standardize the ratings and ensure fair comparisons, we normalized the interaction data using a standard scaler.
- **Item Similarity Computation**: Using the normalized interaction matrix, we computed item-to-item similarities based on cosine similarity, identifying questions that were closely related based on user preferences.

5. Experiments

5.1 Dataset

In our Recommendation System for DSA (Data Structures and Algorithms) Questions, we initially utilized a dataset sourced from Kaggle, which contained 260 rows. While this provided a good starting point, the dataset required significant enhancement to meet the accuracy and diversity needed for effective recommendations.

To achieve this, we manually curated and expanded the dataset by adding data from trusted platforms such as LeetCode, GeeksforGeeks, and Codeforces, ultimately growing the dataset to 740 rows. Additionally, we performed several preprocessing and data processing in the code to ensure data quality and consistency.

The dataset includes the following columns:

1. **Question Name:** The name of the question or problem.
2. **Difficulty Level:** The level of difficulty (e.g., Easy, Medium, Hard).
3. **Total Submissions:** The number of submissions made for the question.
4. **Accuracy:** The success rate of solving the question.
5. **Company Tags:** Tags associated with companies that commonly use the question in their interviews (e.g., Google, Amazon).
6. **Source:** The platform or source of the question (e.g., Kaggle, LeetCode, GeeksforGeeks, Codeforces).

Key steps in our preprocessing included:

1. **Data Cleaning and Handling Missing Values:** Missing values were imputed to ensure completeness and maintain the dataset's integrity.
2. **Data Type Conversion:** Columns such as total Submissions and accuracy were converted from object types to numeric values, enabling better analysis and modeling.
3. **Text Preprocessing:** Problem descriptions were tokenized, converted to lowercase, and filtered to remove stopwords and non-alphanumeric characters, ensuring a clean and consistent dataset.
4. **Label Encoding:** difficulty levels were encoded into numeric values to enable effective modeling and analysis, with each difficulty level assigned a unique integer representation.
5. **Feature Engineering:** A new feature called **Combined_Features** was created to merge various attributes of each problem into a single string. This feature combined the processed problem statement, difficulty level, total submissions, accuracy, and company tags for each problem. The combined feature enhanced the system's ability to analyze and rank problems effectively.

By combining these processes, we significantly improved the dataset's quality, nearly tripling its size while ensuring its reliability and alignment with user needs. The final dataset supports a recommendation system that provides personalized, accurate, and trustworthy suggestions, enhancing the learning experience for users.

5.2 System Specifications

The project was implemented on a system running Windows 11 (64-bit) with an Intel Core i5 processor clocked at 2.3GHz, 16 GB of RAM, and an NVIDIA v 8.1.966.0 GPU.

Python was utilized as the programming language, and important libraries were Matplotlib/Plotly for visualization creation, Pandas for data management, and Scikit-learn for TF-IDF and cosine similarity implementation.

5.3 Results

The goal of this experiment was to develop and evaluate a personalized recommendation system for Data Structures and Algorithms (DSA) questions using **Content-Based Filtering (CBF)**. This approach utilizes **TF-IDF (Term Frequency-Inverse Document Frequency)** to analyze and rank problems based on their textual and metadata features. We combined these features into a unified vector representation, capturing important aspects such as difficulty level, total submissions, accuracy, and associated tags.

Using the **Cosine Similarity** metric, we measured the similarity between questions. Based on these similarities, the system recommended questions that closely matched the selected query. The experiment aimed to evaluate the system's effectiveness by comparing its predicted recommendations to the actual question difficulty levels.

Evaluating Recommendation System Based on Difficulty Level:

	precision	recall	f1-score	support
Easy	0.73	0.33	0.46	33
Medium	0.90	0.97	0.93	540
Basic	0.00	0.00	0.00	8
School	0.82	0.72	0.77	151
Hard	0.44	0.50	0.47	8
accuracy			0.87	740
macro avg	0.58	0.50	0.53	740
weighted avg	0.86	0.87	0.86	740

Fig.1.Classification Report

Overall Metrics:

- Accuracy: **87%**
- Macro Average F1-Score: **0.53**
- Weighted Average F1-Score: **0.86**

Observations:

1. **Strengths**:

- The model performed exceptionally well for **Medium** difficulty questions, achieving an F1-score of **0.93**. This can be attributed to the balanced representation of these questions in the dataset.
- The system effectively leveraged metadata and textual features to recommend relevant questions.

2.Limitations:

- Performance was poor for Basic difficulty questions, with precision, recall, and F1-scores of 0.00. This indicates a severe data imbalance or lack of distinguishing features for these questions.
- Easy questions had moderate precision (0.73) but suffered from low recall (0.33), leading to an F1-score of 0.46.

Ridge Regression

Ridge regression was employed as a classification model to predict the difficulty levels of questions. The model was trained on scaled TF-IDF feature representations and tuned using hyperparameter optimization to achieve the best performance.

Evaluating Ridge Regression Model Based on Difficulty Level:

	precision	recall	f1-score	support
Easy	1.00	1.00	1.00	33
Medium	1.00	1.00	1.00	540
Basic	1.00	1.00	1.00	8
School	1.00	1.00	1.00	151
Hard	1.00	1.00	1.00	8
accuracy			1.00	740
macro avg	1.00	1.00	1.00	740
weighted avg	1.00	1.00	1.00	740

Fig.2.Classification Report

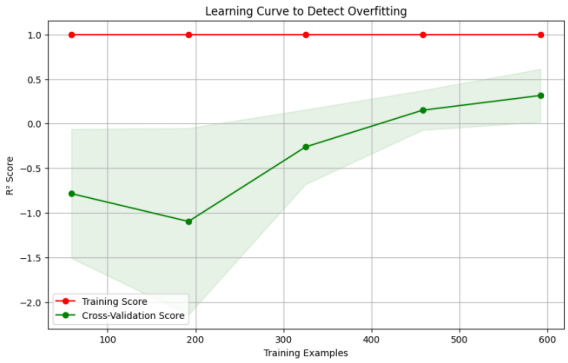

Fig.3.Learning Curve

Overall Metrics:

- Accuracy: **100%**
- Macro Average F1-Score: **100**
- Weighted Average F1-Score: **100**

Observations:

1. **Strengths**:

- Perfect classification performance on the training set.
- Effective in capturing patterns within the dataset
- Highlighted the need for balancing bias and variance through hyperparameter tuning.

2. **Limitations**:

- Overfitting observed in the learning curve, where the model memorized the training data but failed to generalize to new data.
- Limited interpretability of predictions for underrepresented classes (e.g., Basic and Hard).

Hybrid Model CF-CBF

The hybrid recommendation model combines Content-Based Filtering (CBF) and Collaborative Filtering (CF) to enhance personalization. CBF uses metadata like difficulty level, while CF analyzes user interactions, with weights balancing their contributions.

Evaluating Hybrid Recommendation System Based on Difficulty Level:

	precision	recall	f1-score	support
Easy	0.21	0.42	0.28	33
Medium	0.88	0.38	0.53	540
Basic	0.00	0.00	0.00	8
School	0.28	0.81	0.42	151
Hard	0.43	0.38	0.40	8
accuracy			0.47	740
macro avg	0.36	0.40	0.33	740
weighted avg	0.71	0.47	0.49	740

Fig.4.Classification Report

Overall Metrics:

- Accuracy: **47%**
- Macro Average F1-Score: **33**
- Weighted Average F1-Score: **49**

Observations:

1**. Strengths**:

- Flexibility in the weighting parameter (alpha) allows fine-tuning the balance between CBF and CF, enabling customization based on specific datasets or use cases.
- By integrating content-based and collaborative filtering, the hybrid model can overcome the limitations of either approach alone.

3. **Limitations**:

- While the hybrid model addresses the cold start issue for items, it struggles because it does not have the real interactions of the user instead it has the dummy values of the interactions.

6. Discussion

This project presented several challenges that impacted the development and evaluation of our recommendation system. The first major hurdle was the dataset size; the initial dataset contained only 260 questions. To address this, we collected additional data, expanding the dataset to 740 questions. However, even with this expansion, the dataset size remained limited, which constrained the robustness of our models and their ability to generalize.

In evaluating the hybrid model, which combined content-based filtering (CBF) and collaborative filtering (CF), we observed poor results instead of the expected improvement. This underperformance can be attributed to the use of dummy values for user-item interactions, as actual interaction data was not available. With real user interaction data, the hybrid model would likely have produced more meaningful and accurate recommendations.

The machine learning approach, specifically the Ridge regression model, also faced significant challenges. Despite performing hyperparameter tuning and testing different configurations, the model suffered from overfitting. While the training performance was strong, the results on the testing data were suboptimal, highlighting the limitations imposed by the small dataset and its inability to capture sufficient variance for effective model training.

Interestingly, using content-based filtering alone, leveraging TF-IDF and cosine similarity, achieved the best results among all approaches. This demonstrates the effectiveness of extracting insights from item metadata in the absence of robust interaction data.

Lastly, we did not compare the results of our system to the original paper because their dataset was significantly different from ours. This difference in datasets made a direct comparison infeasible, as their dataset structure and size provided advantages that we could not replicate with our dataset.

7. Conclusion

The content-based filtering approach, utilizing TF-IDF and cosine similarity, delivered the best results in this project. However, limitations such as the small dataset size and the use of dummy interaction values negatively impacted the performance of the hybrid and machine learning models. Future efforts should focus on acquiring real user interaction data, expanding the dataset, and aligning with benchmark studies to enable meaningful comparisons and improve recommendation accuracy.

8. References

[1] Karthik, R., & Ganapathy, S. (2021). A fuzzy recommendation system for predicting the customer's interests using sentiment analysis and ontology in e-commerce. Applied Soft Computing, 108, 107396.

[2] Javed, U., Shaukat, K., Hameed, I. A., Iqbal, F., Alam, T. M., & Luo, S. (2021). A review of content-based and context-based recommendation systems. International Journal of Emerging Technologies in Learning (iJET), 16(3), 274–306.

[3] Natarajan, S., Vairavasundaram, S., Natarajan, S., & Gandomi, A. H. (2020). Resolving data sparsity and cold start problems in collaborative filtering recommender systems using Linked Open Data. Expert Systems with Applications, 149, 113248.

[4] Dataset https://www.kaggle.com/datasets/inductiveanks/dsa-questions-dataset

[5] Scientific paper https://www.mecs-press.org/ijisa/ijisa-v15-n5/IJISA-V15-N5-3.pdf