Experiment No: 9

Aim: Mini Project on Designing Recommendation System

<table>
<tr><td colspan="3">Title:</td></tr>
<tr><td rowspan="2">Team Members</td><td>Name: Ishani Bhadlikar</td><td>Sap ID: 60018210025</td></tr>
<tr><td>Name: Yashvi Gogri</td><td>Sap ID: 60018210056</td></tr>
</table>

1. Introduction

In today's digital era, where online learning has become an integral part of education, providing personalized course recommendations is essential to enhance the learning experience. A recommendation system for online courses can address the challenge of navigating the vast array of available courses, helping learners find content aligned with their interests, goals, and skill levels. This project focuses on building a course recommendation system as a mini-project, aiming to improve learning pathways for users and simulate real-world applications in educational technology.

Motivation for the Project : The motivation for creating a recommendation system lies in its potential to make a meaningful impact on the way learners interact with online platforms. With an overwhelming number of courses offered on platforms like Coursera, Udemy, and edX, finding the right course can be a daunting task. By tailoring suggestions to each user's unique needs, a recommendation system simplifies this decision-making process, saving time and enhancing the relevance of the courses displayed.

Real-World Applications : The project simulates scenarios commonly seen in e-learning platforms, where personalized recommendations enhance user engagement, improve learning outcomes, and increase course completion rates. With real-world applications in mind, this project also explores the practical challenges of user data handling, algorithm selection, and performance optimization in recommendation systems.

2. Objective and Scope

Main Objective: To develop a Context-Aware Recommendation System (CARS) for online learning platforms that provides personalized course recommendations by integrating multiple contextual factors. The system leverages:

- Content-Based Filtering using course metadata (e.g., name, description, skills).
- Context Awareness by incorporating user preferences, such as difficulty level, and combining similarity scores with normalized course ratings. This approach ensures that the recommendations are not only relevant but also aligned with the user's learning goals and proficiency level.

Scope of the Project: We have used the Coursera.csv dataset which contains information publicly available on the Coursera website. The columns are: Name, University, Difficulty Level, Rating, Link, Description and Skills.

3. Literature Review / Background

Recommendation systems (RS) have become essential in a variety of applications, from e-commerce and social media to educational platforms, due to their ability to enhance user experience by providing personalized content. Different RS methodologies address specific recommendation challenges, contributing to improved relevance, scalability, and user satisfaction. Here, we discuss prominent RS techniques, their evolution, and key challenges as reviewed in recent studies.

Technique	Notable Projects/ Studies	Applications	Key Contributions	Challenges
Collaborative Filtering (CF)	- **GroupLens Project** (Resnick et al., 1994): Movie recommendations. - **Amazon** (Sarwar et al., 2001): Item-based CF. - **Netflix Prize** (Bennett et al., 2007): Matrix factorization.	- Streaming (Netflix). - E-commerce (Amazon, eBay). - Social media.	- Enhanced similarity calculations. - Scalable to large datasets (e.g., Amazon's precomputing techniques). - Popularized SVD techniques (Netflix Prize).	- Data sparsity. - Cold start for new users/items.
Content-Based Filtering (CBF)	**MovieLens Dataset** (Herlocker et al., 1999): TF-IDF for metadata. - **Educational RS** (Lops et al., 2011): Course recommendations. - **Job RS** (Paparrizos et al., 2011): Job matching.	- Content platforms (IMDb, LinkedIn). - Educational systems.	- Leveraged item metadata for precision. - Incorporated text-processin g techniques (TF-IDF).	- Filter bubbles (limited diversity). - Metadata dependency.
Hybrid RS	- **Netflix** (Gomez-Uribe & Hunt, 2015): Combines CF +	- Multidomain platforms (Netflix, YouTube,	- Improved accuracy by combining methods.	- High computational cost. - Requires

	CBF. - **YouTube** (Covington et al., 2016): Two-stage hybrid. - **Spotify** (Brost et al., 2019): Deep learning + CF.	Spotify).	- Mitigated cold start.	parameter tuning.
Context-Aware RS (CARS)	- **PoiMapper** (Adomavicius et al., 2011): Location-aware POI recommendations. - **Music RS** (Baltrunas et al., 2011): Mood-based playlists. - **E-learning RS** (Li et al., 2017): Personalized course recommendations.	- Travel apps (TripAdvisor). - Music apps (Spotify). - E-learning (Khan Academy).	- Highlighted role of contextual data. - Developed novel contextual modeling methods.	- Sparse contextual data. - Privacy concerns.
Knowledge-Based RS	- **mySugr Diabetes App** (Burke, 2000): Health action recommendations. - **Financial RS** (Jannach et al., 2006): Investment suggestions.	- Healthcare. - Finance. - Specialized domains.	- Used domain knowledge for accurate recommendations. - Rule-based approaches tailored to sensitive domains.	- Domain expertise required. - Limited scalability.

6. Challenges Across RS:

- **Cold Start Problem**: RS often struggle to recommend items for new users or newly added items. Collaborative filtering methods are especially affected by this issue, while content-based approaches may partially address it if item metadata is available.

- **Data Sparsity**: In systems with low user-item interactions, data sparsity limits the effectiveness of collaborative filtering methods. To address this, hybrid systems and matrix factorization techniques have been explored (Koren et al., 2009).

- **Explainability**: As recommendation algorithms grow more complex, providing explainable recommendations becomes crucial to enhance user trust. Various researchers have proposed explainable RS, incorporating techniques like feature visualization or rule-based explanations to clarify recommendations (Zhang & Chen, 2020).

- **Scalability**: With the increasing volume of data in RS applications, scalability is a critical concern. Techniques such as dimensionality reduction and parallel computing have been applied to manage computational complexity and ensure real-time recommendation capabilities.

4. Methodology

<u>Data Collection:</u>

- Dataset Source: Coursera dataset from Kaggle.
- Size and Features:
  - Size: Includes a few thousands of courses.
  - Features: Course Name, Difficulty Level, Course Description, Skills, and Course Rating.
- Pre-processing:
  - Text columns were cleaned by removing special characters and stemming.
  - Created a combined feature column (`tags`) for recommendation.
  - Normalized course ratings to scale between 0 and 1.

<u>Building the Recommendation Model:</u>

- Model Type:
  - **Content-Based Filtering**: Used CountVectorizer to represent course metadata as feature vectors. Calculated cosine similarity between courses for content relevance.
  - **Context Aware Recommender System**: Incorporated course difficulty levels to filter recommendations pre- or post-similarity calculations.Integrated normalized course ratings with similarity scores to improve recommendation quality.
  - Hybrid Scoring: Weighted the similarity score and rating using an adjustable parameter (`alpha`).

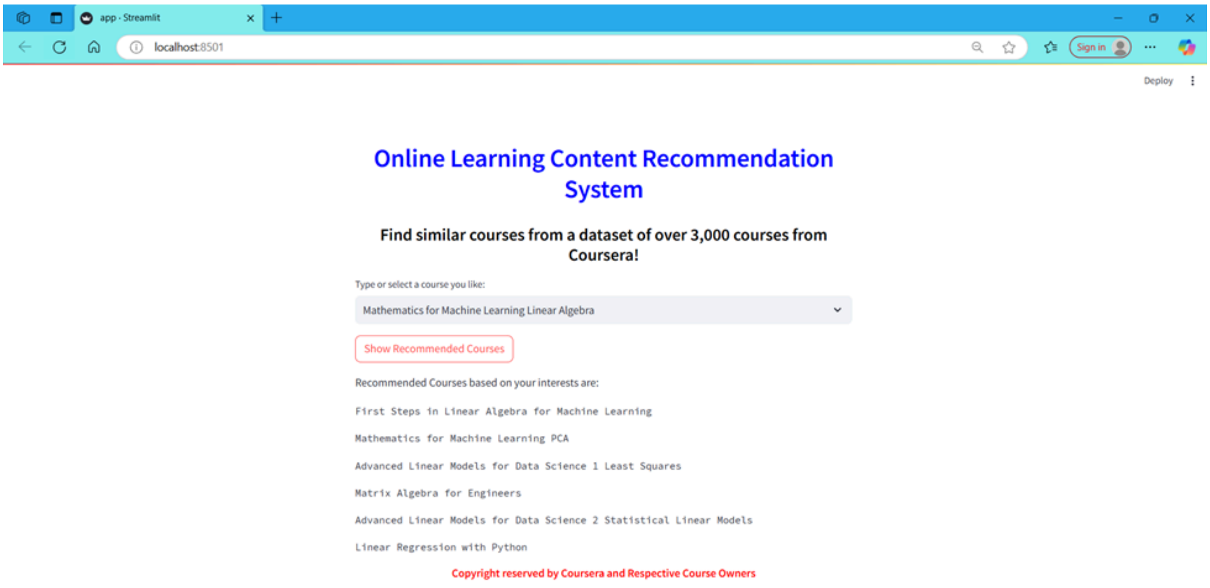
<u>Evaluation:</u>

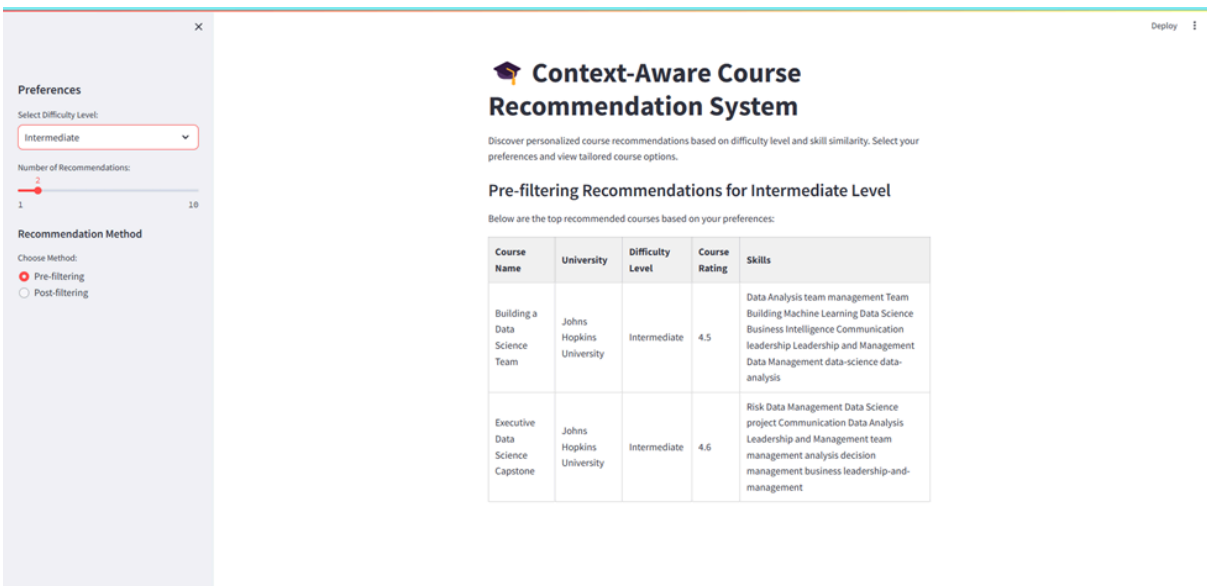
- Metrics: The system's performance was qualitatively assessed based on the relevance of recommendations. Future extensions could use Precision, Recall, and F1-Score for validation.
- Validation: Model quality is validated through manual inspection of recommendations for selected courses. **Train-test split** or **cross-validation techniques** were not directly applied due to the similarity-based approach but could be introduced for future improvements.

Tools and Libraries Used:

- Programming Language: Python.
- Libraries:
  - Data Processing: `pandas`, `numpy`.
  - Natural Language Processing: `scikit-learn`, `nltk`.
  - Similarity Calculations: `scikit-learn` (cosine similarity).
- Tools:
  - Jupyter Notebook for implementation.
  - Pickle for saving model artifacts (e.g., vectorizer, similarity matrix).

5. Results and Analysis (1 page)





6. Conclusion :

<u>Goals, Methods, and Results-</u>

- Goals: Develop a Context-Aware Recommendation System (CARS) to provide personalized course recommendations using content-based filtering and contextual factors like difficulty level and ratings.
- Methods:
  - Cleaned and processed course data into a unified feature (`tags`), vectorized using CountVectorizer.
  - Calculated cosine similarity to rank courses by content relevance.
  - Incorporated contextual filtering (difficulty level) and hybrid scoring (combining similarity and ratings).
- Results: The system effectively provided relevant, personalized recommendations, but relied heavily on textual metadata. Qualitative evaluation demonstrated promising results.

<u>Reflection:</u>

- Success: Integrated multiple contextual factors, improving recommendation relevance and diversity.
- Limitations: Lacked quantitative evaluation metrics, struggled with cold-start problems, and relied on dataset metadata.

<u>Future Work</u>

Model Improvements:

1. Advanced Algorithms: Integrate deep learning techniques (e.g., embeddings or neural collaborative filtering) for better pattern recognition.
2. Cold-Start Problem: Combine collaborative filtering with user profile data to improve recommendations for new users or courses.

3. Evaluation: Use metrics like Precision, Recall, and F1-Score; incorporate A/B testing with real user data.
4. Dynamic Contexts: Add temporal and behavioral contexts for adaptive recommendations.

Applications and Scalability:

- Applications: Enhance e-learning platforms like Coursera by offering personalized course discovery.
- Scalability: Use cloud-based infrastructure and distributed computing for real-time, large-scale deployments.

7. References

1. https://www.kaggle.com/code/sagarbapodara/coursera-course-recommendation-system-webapp/input
2. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). *GroupLens: An Open Architecture for Collaborative Filtering of Netnews*. Proceedings of the 1994 ACM conference on Computer supported cooperative work, 175–186.
3. Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). *Item-Based Collaborative Filtering Recommendation Algorithms*. Proceedings of the 10th International Conference on World Wide Web, 285–295.
4. Pazzani, M. J., & Billsus, D. (2007). *Content-Based Recommendation Systems*. In The Adaptive Web (pp. 325–341). Springer.
5. Gomez-Uribe, C. A., & Hunt, N. (2015). *The Netflix Recommender System: Algorithms, Business Value, and Innovation*. ACM Transactions on Management Information Systems (TMIS), 6(4), 1–19.
6. Burke, R. (2002). *Hybrid Recommender Systems: Survey and Experiments*. User Modeling and User-Adapted Interaction, 12(4), 331–370.
7. Adomavicius, G., Sankaranarayanan, R., Sen, S., & Tuzhilin, A. (2005). *Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach*. ACM Transactions on Information Systems, 23(1), 103–145.
8. Li, J., Xu, L., & Wang, W. (2017). *Context-Aware Recommendations in E-Learning Systems*. IEEE Access, 5, 16301–16314.
9. Shi, Y., Larson, M., & Hanjalic, A. (2014). *Collaborative Filtering beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges*. ACM Computing Surveys (CSUR), 47(1), 1–45.
10. Zhang, Y., & Chen, X. (2020). *Explainable Recommendation: A Survey and New Perspectives*. Foundations and Trends in Information Retrieval, 14(1), 1–101.
11. Burke, R. (2000). *Knowledge-Based Recommender Systems*. Encyclopedia of Library and Information Systems, 69(Supplement 32), 180–200.
12. Koren, Y., Bell, R., & Volinsky, C. (2009). *Matrix Factorization Techniques for Recommender Systems*. Computer, 42(8), 30–37.