FORMAN CHRISTIAN COLLEGE
(A CHARTERED UNIVERSITY)

Department of Computer Science
Forman Christian College University

COMP360: Introduction to AI
Spring 2025

Final Project Report

AI-Based Show Recommender System

Team Members and Roles

· **Member 1:** Ayan Ali - 261943131@formanite.fccollege.edu.pk

· Role: Algorithm development / testing

· **Member 2:** Sheril Bobby - 261939549@formanite.fccollege.edu.pk –

· Role: Data preprocessing and final testing

· **Member 3:** Muhammad Faizan Ahmad - 261945342@formanite.fccollege.edu.pk –

· Role: Interface Development / Quality Assurance

1. Introduction, Objectives, and Problem Statement

Introduction:
In the age of digital entertainment, users are flooded with thousands of TV show options. Finding similar shows to one's liking can be challenging without intelligent tools. Our system recommends K-Dramas based on content similarities, helping users make informed decisions.

Objectives:

- Build a content-based K-Drama recommender using metadata.
- Use a combination of KNN (on tags) and cosine similarity (on content/genres).
- Develop an interactive UI with Streamlit.
- Implement a robust recommendation ranking system.

Problem Statement:
The challenge lies in the overwhelming volume of available entertainment content, making it difficult for users to find shows aligned with their preferences. The proposed recommendation system addresses that by giving show recommendations based on the similarity index of the show given by the user.

2. Detailed Methodology

Architecture Overview:

- The user selects a show.
- We extract TF-IDF vectors for tags, content, and genres.
- Use **K-Nearest Neighbors (KNN)** for tag similarity.
- Compute **cosine similarity** for content and genre.
- Combine all three using weighted scoring:
  - Tags: 60%
  - Genre: 30%
  - Content: 10%

Workflow:

1. Clean data using Pandas and regex.
2. Use TfidfVectorizer for textual feature extraction.
3. Apply NearestNeighbors for tag-based KNN.
4. Calculate cosine similarity matrices for genres and content.
5. Compute a weighted average of similarities.
6. Display the top 10 similar shows.

3. Dataset Description and Preprocessing

Dataset Used:

- **Title:** Top 5000 Popular Drama Details
- **Source:** mydramalist.com (via Kaggle)
- **Fields Used:** name, genres, tags, content, rating

Preprocessing Steps:

- Filled missing values in genres, tags, and content.
- Cleaned tags using regex (re.sub(r'[/]', ' ', tag)).
- Converted all text to lowercase for uniformity.
- Created a unified representation of textual metadata.

4. Experiments Conducted

In version 1.0, a content-based recommender was built using TF-IDF and cosine similarity on combined features—content, genres, and tags. To emphasize tags, their weight was increased by repeating them. This simple model recommended similar shows based on overall text similarity without separating feature importance explicitly.
Problems:

- No distinction between features (tags, genres, content treated equally).
- Tag importance was artificially boosted by repeating them, which is a hacky weighting method.
- Less control over how much each feature contributes to recommendations.

In version 1.1, content, genres, and tags were **vectorized separately** using TF-IDF and their individual **cosine similarities** were calculated. A **weighted combination** of these similarities was used to recommend shows, offering more control over the influence of each feature. This approach improved flexibility and interpretability compared to the first version.
Problems:

- Calculated similarity across all pairs, making it less efficient (especially with large datasets).
- No filtering or narrowing down the candidate shows before similarity scoring.
- Still treated tags as plain text without preprocessing or phrase handling.

In the version 2.0, the recommender system used K-Nearest Neighbors (KNN) on genre-based TF-IDF vectors to first filter potentially similar shows. Then, it calculated cosine similarity for tags and content, combining them using weighted scores. This hybrid approach improved efficiency and relevance by narrowing the comparison set before deep similarity analysis.
Problems:

- Genres used for initial filtering, but tags proved to be more relevant in case of kdramas.

- Tags were not preprocessed, so multi-word or slash-separated tags were not handled well.
- No user interface for interaction.

the fourth,the last and the best performing version 3.0, the recommender system was deployed as an interactive Streamlit web app. It applied preprocessing to tags by replacing slashes and splitting phrases to better capture multi-word tag meanings. Similar shows were identified using KNN on cleaned tag data, followed by cosine similarity on content and genres. A weighted combination of all three features provided accurate and user-friendly recommendations.

In version 3.1 and 3.2, we also tried adding country weights and country specific weights but the precisions decrease so we decided to go with the original version 3.0.

Version	**Key Approach**	**Improvements Over Previous**	**Problems / Limitations**
1.0	TF-IDF + cosine similarity on combined content, genres, and repeated tags	Simple baseline	No feature separation, hacky tag weighting, no control over feature influence
1.1	Separate TF-IDF + cosine for content, genres, tags with weighted similarity	More control & interpretability	Inefficient (all-pairs similarity), no filtering, basic tag processing
2.0	KNN on genre vectors → filtered candidates → cosine on tags & content	Improved efficiency & relevance	Genre filtering less relevant (tags better), unprocessed tags, no user interface
3.0	Streamlit app; KNN on cleaned tags → cosine on content & genres (weighted)	Preprocessing tags, interactive UI	None major; most effective version
3.1 / 3.2	Added country/country-specific weights	Tried localization	Precision decreased; reverted to version 3.0

#code for all the trials and version can be found in trials folder

Parameter Tuning:

Different parameter values were tested to get the best possible results, In case of k-drama shows it was found that tags were most relevant for similarity and then the content (description) and genre were second most important.

With k = 100 : Precision dropped

Decreasing the genre weight made the results too diverse to be relevant.

Optimal Parameters found :

Number of Neighbors (K): 50 (for max intralist similarity), filtered to top 10
Weighting Scheme:
Tag Weight: 0.6
Genre Weight: 0.30
Content Weight: 0.10

5. Evaluation Metrics for Content-Based Filtering

Qualitative Evaluation and Percision@K:

Precision@K is a key evaluation metric for recommender systems that measures the proportion of relevant items among the top K recommendations provided to the user. In the context of our content-based K-Drama recommender, K was set to 10, meaning we assessed the top 10 shows suggested by the system for each input title.

To determine relevance, we manually evaluated whether each recommended show shared significant attributes with the input show. These attributes included similar tags (e.g., "supernatural", "romance", "time travel"), comparable settings(such as historical eras, school environments, or workplace dramas), and genre alignment (like mystery, fantasy, comedy, or melodrama).

Based on this manual evaluation, our system achieved an average **Precision@K of 7/10,** indicating that approximately 70% of the recommendations were thematically or structurally relevant to the show selected by the user. This precision score reflects the model's ability to accurately identify shows that resonate with the original show's core themes, tone, or narrative elements.

For example, when the user selected a show like "Goblin", the system successfully recommended titles featuring supernatural beings, emotionally rich plotlines, and romantic-fantasy narratives. These high-relevance matches illustrate the strength of our combined similarity approach (tag-based KNN and cosine similarity on genres and content).

However, some recommendations were only partially aligned, such as sharing a setting or mood but lacking the central genre or theme of the original. In a few cases, a recommendation was deemed irrelevant due to either weak metadata or insufficient tag preprocessing. These limitations offer insight into areas where the model can be further refined.

Overall, our **7/10 Precision@K** reflects a strong performance for a purely content-based system without user feedback loops and implementation of neural networks. It suggests that the model is effective in surfacing shows that are not only similar in superficial tags but also share deeper thematic or genre-related connections.

B. Intralist Similarity

Checks whether the recommended items are coherent (i.e., similar to each other)

The highest intralist similarity index observed during testing was **0.1152**

And the lowest was **0.0651**

This low number indicates that although our list entries were similar to the given show, they were not very similar to each other, the main reason observed was that each list entry show has a specific plot aspect that is similar to the given show and thus the 10 recommended contained shows that that had different aspects similar to the given show.

This tells you how tight or loose your recommendation cluster is.

C. Diversity

Low intralist similarity also means that although show recommended in our list were similar in some aspects,but diverse in other aspect which keeps the user from getting bored by watching identical plotlines

Metric	Purpose	Notes
Cosine Similarity	Measures vector closeness	Applied to genre and content vectors
User Relevance Check	Manual check of top 10 shows	Results were highly relevant
Speed & UI Response	Measured using @st.cache_data	Instant responses (~1s latency)

6. Ethical Considerations

- Dataset does not involve personal data — only metadata.

- Ensured fairness by not filtering shows based on popularity or rating.
- The model does not store or track user inputs or preferences.

7. Key Learnings and Reflections

- **Multimodal Similarity** improves recommendation quality significantly.
- Streamlit is a powerful and easy tool for deploying AI prototypes.
- Weighted scoring allowed control over model behavior.
- TF-IDP.

8. Learning log :

- In Version 1.0: We learned that boosting tag importance by repetition was a quick solution but lacked control over feature contribution.

- In Version 1.1: We learned that vectorizing features separately and combining them with weights improved flexibility and control.

- In Version 2.0: We learned that using **KNN** for initial filtering improved efficiency, but tag preprocessing was necessary for better accuracy in Kdramas.

- In Version 3.0: We learned that deploying an interactive Streamlit app and preprocessing tags allowed for more accurate recommendations and better user experience.

- In Version 3.1 & 3.2: we learned that adding country-specific weights lowered precision, so we reverted to the original version 3.0 for better performance.

- While building a show recommendation app, we learned about **TF-IDF** (Term Frequency-Inverse Document Frequency). It helped identify important words in show descriptions by comparing how unique they were across all shows. This made the recommendations more accurate by matching shows with similar content.

8. Reference

Data set :
https://www.kaggle.com/datasets/anittasaju/complete-5000-dramas-from-mydramalist-review-info
Tools :
https://pandas.pydata.org/
https://scikit-learn.org/stable/
https://streamlit.io/