Project Report

Which Anime should you watch next?

Aafaf RIDOUAN
Ismail CHAKRANE

2024 - 2025

Contents

1 - Introduction

Through this project, we aim to address the challenge of recommending a new item (anime) to a user based on their interaction history with similar items. Specifically, we framed our research question as: **Which Anime should you watch next?** This focus aligns with our context, which centers on anime recommendations.

1.1 Dataset Description

The dataset used in this project was obtained from the Kaggle website. It is sourced from `myanimelist.net` platform through their API. The dataset consists of two files: `Anime.csv` and `Rating.csv`, detailed as follows:

1.1.1 `Anime.csv`

This file contains information about various anime and includes the following attributes:

- `anime_id`: Unique identifier for each anime on `myanimelist.net`.
- `name`: The full name of the anime.
- `genre`: A comma-separated list of genres associated with the anime.
- `type`: The format of the anime, such as TV, Movie, OVA, etc.
- `episodes`: The number of episodes in the anime (set to 1 for movies).
- `rating`: The average rating of the anime, scored out of 10.
- `members`: The number of community members associated with the anime's group.

1.1.2 `Rating.csv`

This file contains user-specific ratings for various anime and includes the following attributes:

- `user_id`: A randomly generated, non-identifiable user identifier.
- `anime_id`: The identifier of the anime that has been rated.
- `rating`: The score assigned by the user, rated out of 10. A value of -1 indicates that the user watched the anime but did not assign a rating.

1.2 Organization of the report

In this report, we begin with an **Exploratory Data Analysis (EDA)** phase, where we examine the dataset to extract meaningful insights and identify necessary preprocessing steps. This is followed by the development of two types of recommendation systems: a **Content-Based Recommendation System** and a **Collaborative Filtering Recommendation System**. Both systems are thoroughly explained in subsequent sections.

For each recommendation system, we explore the application of methodologies studied in the course to our dataset. Additionally, we draw conclusions regarding the limitations of these systems and propose potential improvements to enhance their performance.

Finally, the report concludes with a detailed exploration of the **contributions** made in each section of the project. We also provide guidance on how to replicate the implemented solutions within a **development environment**.

2 - Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a critical step in understanding the structure, relationships, and key patterns in a dataset. In this chapter, we analyze the anime dataset to extract insights into its features, distributions, and correlations.

2.1 Preprocessing

Preprocessing is a crucial step in preparing the dataset for analysis, ensuring that the data is clean, consistent, and ready for further exploration. The following preprocessing techniques were applied to the anime dataset:

- **Handling Missing Values:** Missing values in the `genre` column were replaced with empty strings, allowing for consistent analysis of genres.
- **Encoding Categorical Features:**
  - The `genre` column was processed using one-hot encoding to create binary columns for each unique genre.
  - The `type` column was encoded numerically using a label encoder, transforming categories like `TV` and `OVA` into numerical values.
- **Exploding Genre Lists:** To analyze individual genres, the `genre` column was exploded, converting each anime's genre list into multiple rows for better granularity.
- **Handling Unknown Values:** For columns such as `episodes`, unknown values were removed since their ratio was small compared to the total amount of data in the dataset

These preprocessing techniques ensured that the dataset was clean and transformed appropriately for the subsequent analysis (building the recommendations systems).

2.2 Genre Distribution

The distribution of genres in the anime dataset provides insights into the preferences and trends within the anime community. Figure 2.1 illustrates the frequency of each genre across all animes.

From the chart, we observe the following key insights:

- **Comedy**: The most frequent genre, appearing in over 4,000 animes. This highlights the widespread appeal of humor in anime, catering to diverse audiences.
- **Action, Adventure, and Fantasy**: These genres are also highly prevalent, indicating their strong connection to mainstream and popular anime themes. They often overlap with other genres to create engaging and dynamic storytelling.
- **Niche Genres**: Genres such as *Cars*, *Josei*, *Shounen Ai*, and *Yuri* appear less frequently, reflecting their more specialized audience base.
- **Broad Coverage**: The dataset captures a wide variety of genres, from lighthearted categories like *Comedy* to intense themes like *Horror* and *Psychological*.

The genre distribution demonstrates the diversity of content available in anime, catering to a wide range of tastes and preferences. While popular genres like *Comedy* and *Action* dominate, the presence of niche genres suggests that the anime industry also serves specialized audiences, enriching the medium with a variety of unique experiences.

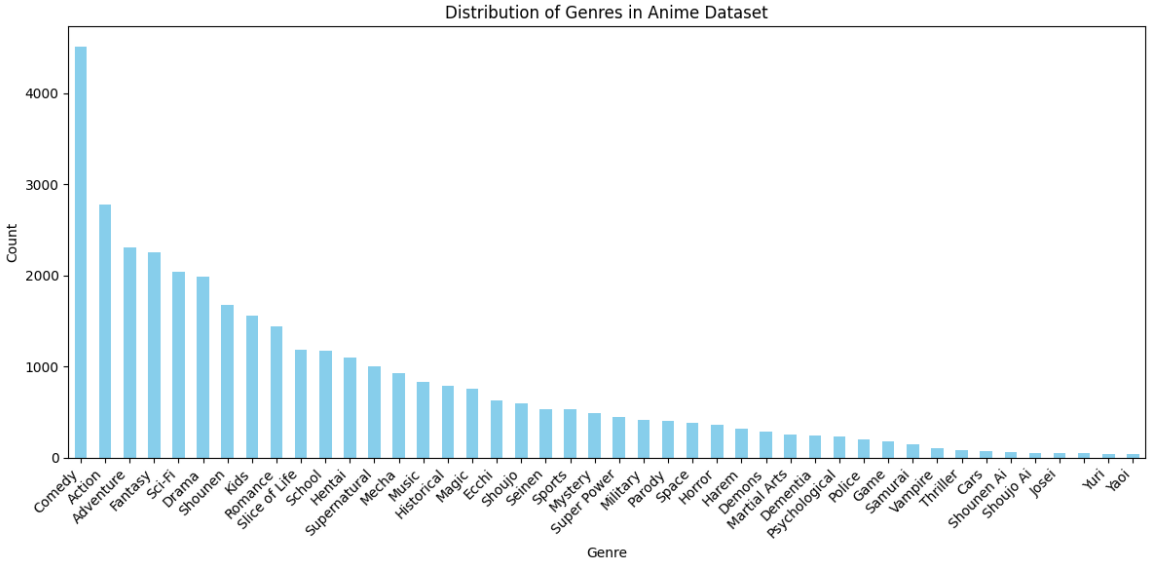

Figure 2.1: Distribution of Genres in the Anime Dataset

2.3 Anime Popularity

Popularity is a key metric for analyzing user engagement with animes. This section examines two aspects of anime popularity: the number of ratings received and the average rating. These aspects are visualized in Figures 2.2 and 2.3, respectively.

2.3.1 Most Popular Animes by User Rating Count

Figure 2.2 highlights the 30 most popular animes based on the number of user ratings. The chart shows that *Death Note* leads significantly, followed by other widely acclaimed titles such as *Sword Art Online* and *Attack on Titan (Shingeki no Kyojin)*. These animes belong to genres like *Action*, *Adventure*, and *Fantasy*, which are typically associated with high user engagement. Additionally, most of the animes in this list are well-known mainstream titles, indicating their widespread appeal across audiences.

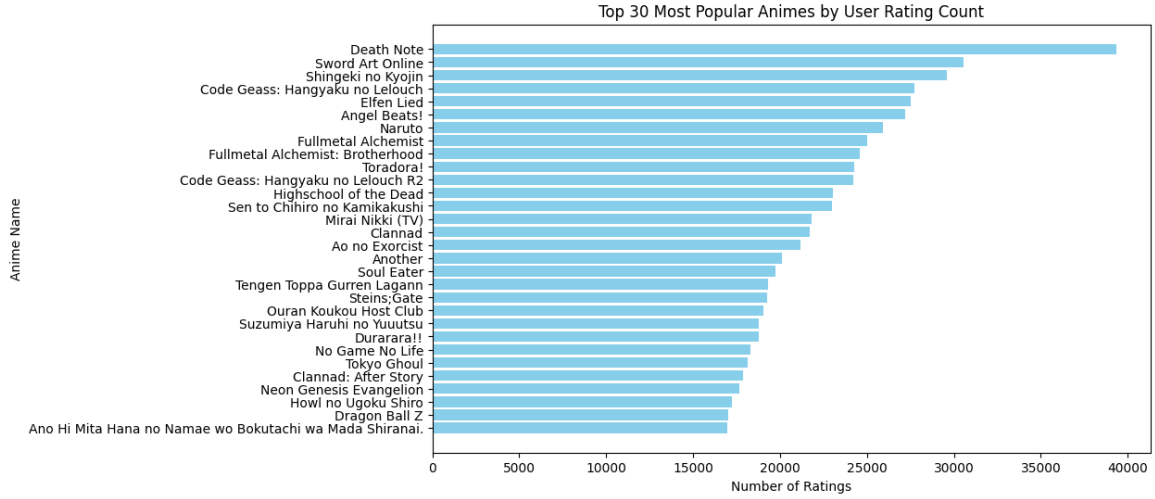

Figure 2.2: Top 30 Most Popular Animes by User Rating Count

2.3.2 Top Animes by Average Rating

Figure 2.3 shows the 30 highest-rated animes based on their average user ratings. Interestingly, these animes are relatively less known and consist largely of specialized or niche titles. Examples include *Kamiusagi Rope x Panasonic Collaboration* and *What's Michael?*, which cater to specific audiences rather than the general anime-viewing population. This suggests that high average ratings are often observed in animes with a smaller, more focused audience base who rate them highly due to personal preference or strong engagement within the niche.

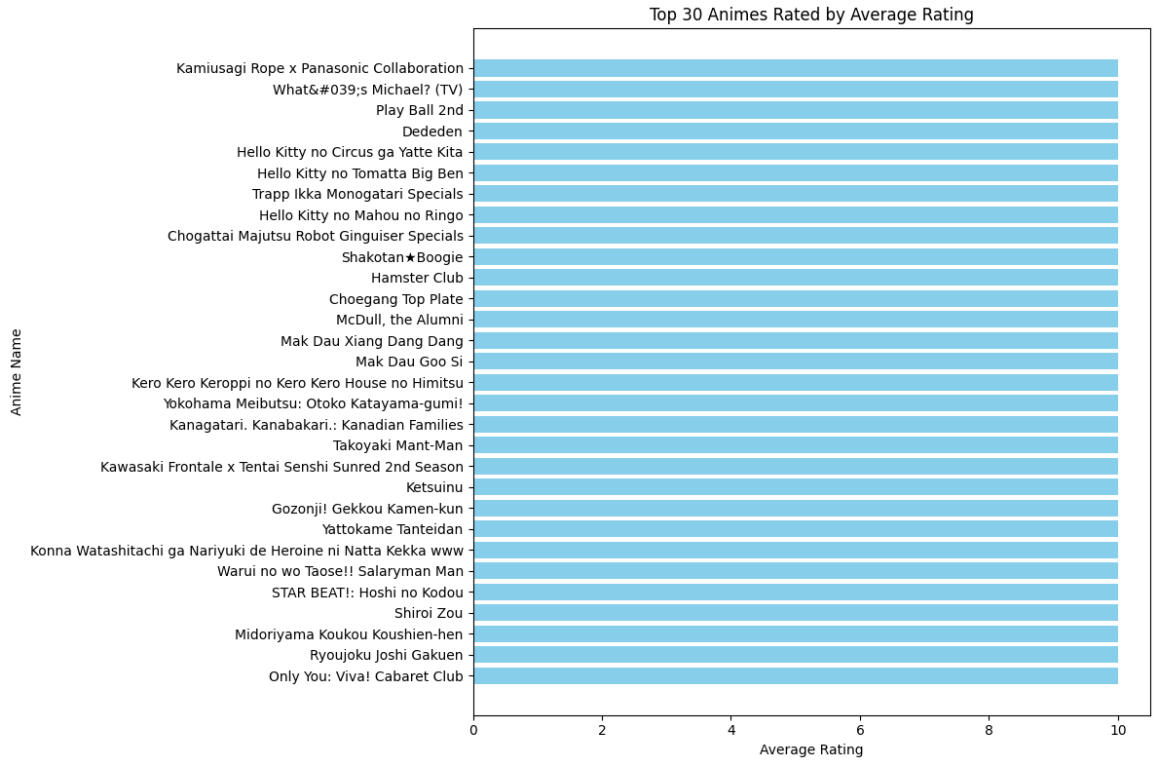

Figure 2.3: Top 30 Animes Rated by Average Rating

The comparison of Figures 2.2 and 2.3 reveals a clear distinction between the two measures of popularity:

- **User Rating Count:** Animes with high user ratings tend to be mainstream, widely marketed, and accessible to a broad audience. They often belong to popular genres and have significant cultural impact, leading to a large number of ratings.
- **Average Rating:** Highly-rated animes tend to be niche, with a smaller but highly engaged audience base. These animes may not have widespread recognition but achieve high ratings due to the dedication of their viewers.

2.4 Rating Distribution

The distribution of user ratings was studied to understand user preferences. A histogram of ratings (Figure 2.4) showed that most ratings lie between 7 and 9, indicating a positive skew in user reviews.

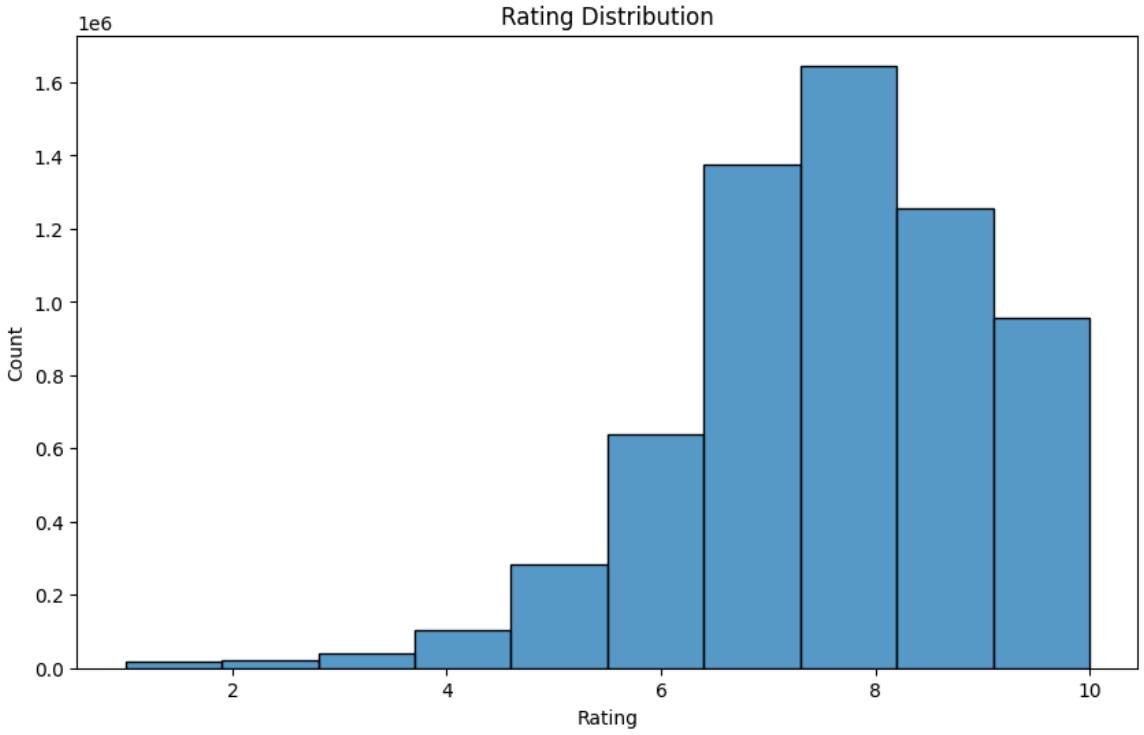

Figure 2.4: Distribution of User Ratings

2.5 Anime Features Correlations

Analyzing correlations between features provides valuable insights into how various attributes of animes are related. This section explores the relationships between features such as `type`, `episodes`, `rating`, `members`, and individual genres. The analysis was performed using a correlation matrix, which quantifies the strength and direction of relationships between features.

The following preprocessing steps were applied to prepare the data for correlation analysis:

- **Filtering Rows:** Removing Animes with `Unknown` values in the `episodes` to ensure numerical consistency.
- **One-Hot Encoding:** The `genre` column, which contains comma-separated values, was split into individual genres, and binary columns were created for each genre using one-hot encoding.
- **Label Encoding:** The `type` column was encoded into numerical values using a label encoder. Categories like `TV` and `Movie` were assigned unique numerical identifiers.
- **Feature Combination:** The one-hot encoded genre columns were concatenated with other numerical features (`type_encoded`, `episodes`, `rating`, and `members`) to form a comprehensive feature set.

The correlation matrix quantifies the relationships between features, with values ranging from -1 (perfect negative correlation) to $+1$ (perfect positive correlation). Figure 2.5 illustrates the computed correlations.

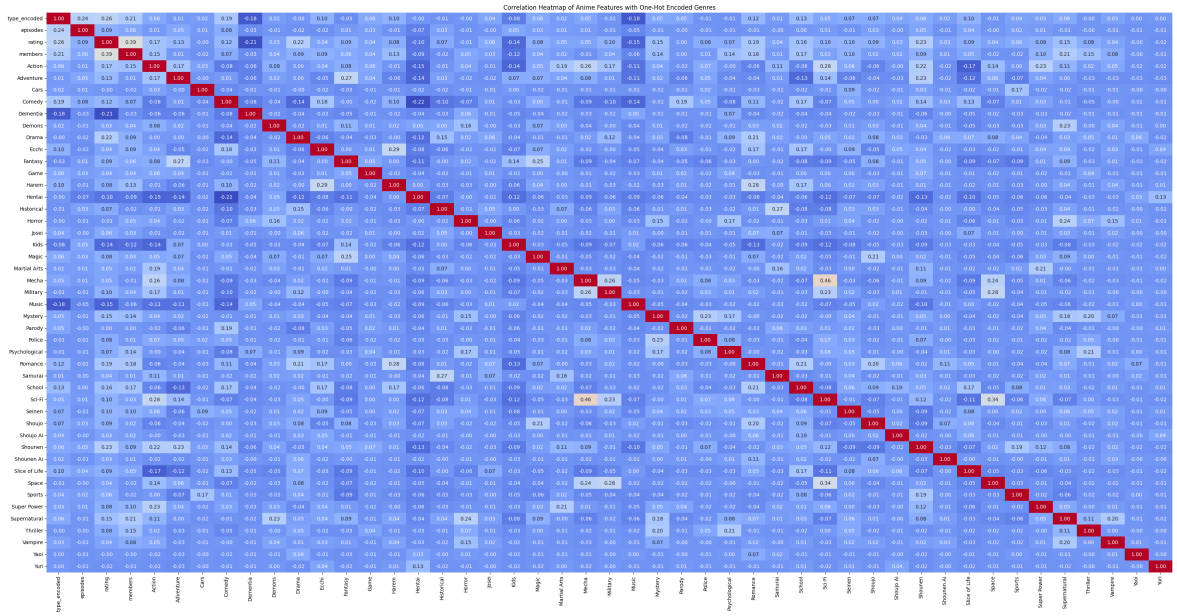

Figure 2.5: Correlation Heatmap of Anime Features with One-Hot Encoded Genres

From the correlation heatmap in Figure 2.5, the following observations can be made:

- **Limited Strong Correlations:** There is no strong positive correlation (close to +1) observed between most features, indicating that the dataset contains diverse and independent relationships among attributes.
- **Genre Relationships:**
  - Certain genres, such as *Action* and *Adventure*, show moderate positive correlations with the number of members, reflecting their appeal to a broader audience.
  - Niche genres, such as *Josei* or *Psychological*, exhibit weaker correlations with the audience size, suggesting their targeted viewer base.
- **Type Differences:** Differences in correlation between `type_encoded` and other features (e.g., `rating`) reveal varying audience reception across anime types (e.g., `TV` vs. `Movie`).

2.6 Conclusion

The exploratory data analysis provided key insights into the dataset:

- Comedy, Action, and Adventure are the most common genres.
- Most user ratings fall between 7 and 9, indicating a positive bias.
- The popularity of animes varies based on ratings, reflecting diverse niche preferences within the broader anime culture.
- There is no strong correlation between the features, leading us to include all features in our content-based recommendation system.

3 - Content-Based Recommendation System

Content-based recommendation serves as the foundation for addressing our core question: "which anime should be recommended to a user to watch next?" The idea behind collaborative filtering is to leverage the features of animes to recommend new ones. The recommended animes are ranked based on a similarity score, calculated between the features of the original anime and those of the recommended animes.

3.1 Data Preparation

As outlined in the notebook `rec_sys_content.ipynb`, which contains the code for this section, we began by cleaning the names of animes by removing special characters. Since the data was sourced from the web, some formatting used in HTML was present in the names, such as `&#039`; these special characters were removed.

For entries with missing genres, we added an empty string as a placeholder. Additionally, we dropped a portion of the data that was deemed unnecessary due to its smaller size, as previously identified during the EDA section.

Following this, we combined all features of animes into a single string. This approach is crucial because user preferences can vary; some users might not only consider the genre of animes but also take into account other factors such as the number of episodes, type, and ratings. For instance, a user who prefers shorter, related animes like *Shingeki no Kyojin* (Attack on Titan) might not be interested in longer series such as *One Piece* or *Bleach*, and vice versa.

3.2 Building the Recommendation System

To construct our content-based recommendation system, we utilized the `TfidfVectorizer` from Scikit-learn to transform anime features into numerical representations and the `sigmoid_kernel` function to compute similarity scores between these representations. Below, we explain the process step-by-step.

3.2.1 Vectorization Using TF-IDF

We started by creating a TF-IDF matrix, The `TfidfVectorizer` was initialized with specific parameters:

- `min_df=3`: Ignore terms that appear in fewer than 3 documents.
- `max_features=None`: No limit on the number of features.
- `strip_accents='unicode'`: Normalize text by stripping accents.
- `analyzer='word'`: Tokenize text at the word level.
- `token_pattern=r'\w{1,}'`: Keep tokens of at least one character.
- `ngram_range=(1, 3)`: Include unigrams, bigrams, and trigrams.
- `stop_words='english'`: Remove common English stop words.

3.2.2 Computing Similarity Scores

Next, we calculated the similarity between all pairs of animes using the sigmoid kernel. The `sigmoid_kernel` function computes similarity scores, which we used to rank recommendations.

3.2.3 Recommendation Function

We implemented a content-based recommender function, `content_based_recommender`, to return the top 10 similar animes for a given title. The function operates as follows:

- Maps the anime title to its index using a pandas `Series`.
- Retrieves similarity scores for the given anime.
- Sorts the similarity scores in descending order.
- Extracts the indices of the top 10 most similar animes.
- Returns a DataFrame with the top 10 recommendations, including information such as name, rating, number of members, episodes, genre, and type.

As an example, we selected an anime from the dataset and retrieved recommendations. The results are as follows:

Recommendations for: Choujin Locke: Lord Leon genre:Action, Sci-Fi, Super Power type:OVA episodes:3 rating:6.37 members:483

	Anime name	Rating	Members	Episodes	Genre	Type
0	Hoshi no Ko Chobin	6.36	231	26	Adventure, Sci-Fi	TV
1	Diabolus: Kikoku	6.36	3250	2	Hentai	OVA
2	Shin Hitou Meguri	6.36	2702	2	Hentai	OVA
3	Wonder 3 Pilot	6.36	63	1	Action, Adventure, Comedy, Sci-Fi	Special
4	Last Exile: Ginyoku no Fam Recaps	6.56	4621	2	Action, Adventure, Sci-Fi	Special
5	Wakakusa Monogatari: Nan to Jo-sensei Specials	6.36	138	2	Drama, Historical, School, Slice of Life	Special
6	Subarashii Sekai Ryokou: Alaska no Tabi "Daigo...	8.00	62	2	Adventure, Sci-Fi	Special
7	Gatchaman Crowds: Embrace	7.50	10757	1	Adventure, Sci-Fi	Special
8	Cele Kano	6.36	1947	2	Hentai	OVA
9	Imouto de Ikou!	6.36	2457	2	Hentai	OVA

Figure 3.1: An example of a Content-Based Recommendation

3.3 Evaluation of the Recommendation System using Clustering Techniques

In the absence of a specific ground truth to evaluate the recommendation system within this dataset, alternative techniques were employed to assess its quality. Specifically, clustering methods were utilized to group animes based on their features, and the quality of recommendations was evaluated by verifying whether the recommended animes belonged to the same cluster as the initial anime. Two clustering algorithms were used for this purpose: DBSCAN and Agglomerative Clustering, both employing a cosine-based metric for distance computation.

The choice of these two techniques was not random. In fact, with a TF-IDF feature matrix containing 11,954 animes, we could not opt for K-means clustering because determining the optimal number of clusters is not straightforward. Although we tested some techniques, such as using the silhouette score to find the number of clusters, we abandoned this approach due to its computational complexity. Instead, we chose methods that implicitly determine the number of clusters (K).

3.3.1 DBSCAN Clustering

The DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm was applied to the TF-IDF feature matrix (`tfv_matrix`) of the animes dataset. Using a cosine-based metric for distance computation, DBSCAN effectively identified dense clusters of similar animes. The following steps were undertaken:

1. The TF-IDF matrix was clustered using DBSCAN with parameters $\epsilon = 0.5$ and a minimum of 5 samples per cluster.
2. The clusters generated by DBSCAN were visualized using PCA (Principal Component Analysis) to reduce the high-dimensional TF-IDF matrix to two dimensions.
3. Cluster labels were assigned to the dataset and stored as a new column, `dbscan_cluster`, for validation purposes.

The resulting clusters were visualized as shown in Figure 3.2.

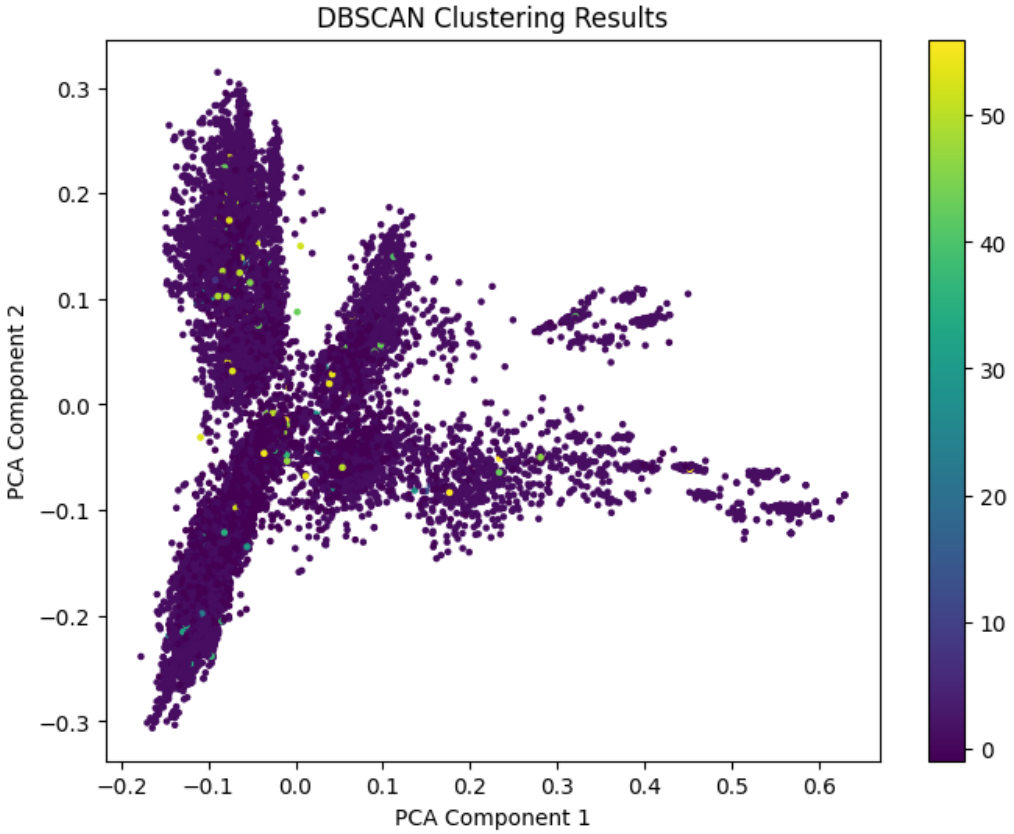

Figure 3.2: DBSCAN Clustering Results Visualized in 2D Space

3.3.2 Agglomerative Clustering

Agglomerative Clustering, a hierarchical clustering method, was also applied to the same TF-IDF matrix. The average linkage criterion with a cosine-based metric was used. The steps were as follows:

1. The TF-IDF matrix was converted to a dense format to enable compatibility with the algorithm.

2. Agglomerative Clustering was applied, producing hierarchical groupings of animes.
3. The high-dimensional feature space was reduced to two dimensions using PCA for visualization.
4. Cluster labels were assigned to the dataset and stored in the `agg_cluster` column.

The resulting clusters are visualized in Figure 3.3.

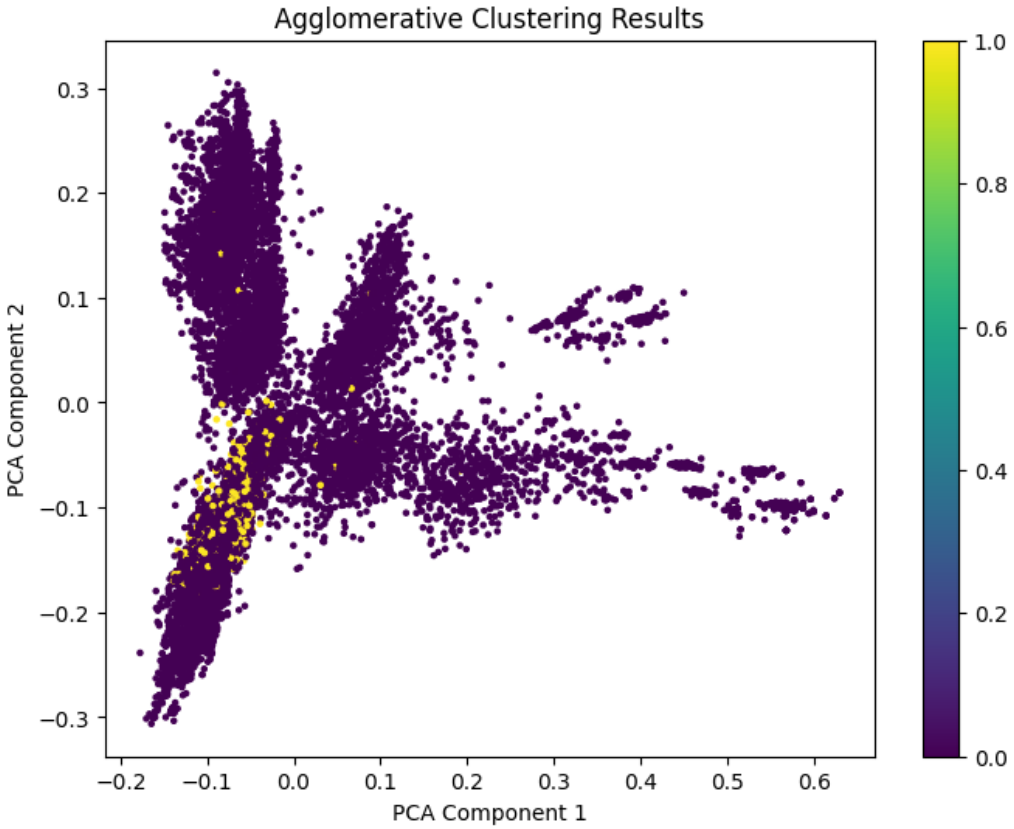

Figure 3.3: Agglomerative Clustering Results Visualized in 2D Space

Our objective here was not specifically to plot the DBSCAN or Agglomerative clusters; these plots were used merely for illustration. The main goal was to utilize these two methods to determine whether our content-based recommendation system performed well.

3.3.3 Validation Methodology

To evaluate the recommendation system, a validation function was implemented to compute *Cluster Purity (%)*. This metric quantifies the percentage of recommended animes that belong to the same cluster as the initial anime. The procedure is as follows:

1. For a selected anime (e.g., *Choujin Locke: Lord Leon*), its cluster label was retrieved from the respective clustering results (`dbscan_cluster` or `agg_cluster`).
2. Recommendations were generated using the content-based recommendation system.
3. The cluster labels of the recommended animes were compared to the cluster label of the selected anime.
4. The percentage of recommended animes in the same cluster was calculated as *Cluster Purity*.

3.3.4 Results

The validation results for the anime *Choujin Locke: Lord Leon* are summarized in Table 3.1.

Clustering Method	Recommendations in the Same Cluster	Cluster Purity (%)
DBSCAN	10	100.0
Agglomerative Clustering	10	100.0

Table 3.1: Validation Results for the Anime *Choujin Locke: Lord Leon*

The results indicate that the recommendations generated by the system are highly consistent with the clusters identified by both DBSCAN and Agglomerative Clustering. This demonstrates the effectiveness of the recommendation system in grouping and recommending similar animes.

4 - Collaborative Filtering Recommendation System

Collaborative filtering has been found to be more efficient than content-based filtering, particularly when dealing with vast catalogs and diverse user preferences. This type of recommendation system relies on user interaction data without requiring detailed item descriptors. It can also uncover complex, latent relationships between users and items that are not easily captured through content-based methods, making it more effective in dynamic, large-scale environments. In this section, we will explore the various types of collaborative filtering recommendation systems.

4.1 Memory Based Collaborative Filtering

In this type of collaborative filtering, we rely on computing the similarity between items or users using a user-item matrix where the rows represent users and the columns represent items (anime in this case), with the ratings as entries. In the following sections, we will implement both user-based and item-based collaborative filtering approaches.

4.1.1 User-based Collaborative Filtering (UBCF)

User-Based Collaborative Filtering (UBCF) involves recommending new anime to a user based on unseen anime rated by other users who are found to be similar to them. This approach focuses on studying the correlation between users to identify those with similar preferences. We start by computing the user-item matrix which is a pivot matrix where the rows represent users and the columns represent animes, with the ratings as entries. We then center each user's ratings around 0 by subtracting the average rating for each user and finally fill the NaN values with zeros (Even though this might affect the results, we couldn't find another way to handle missing values). After this, we compute the correlation between the users to get top N similar users, we predict ratings for animes the target user has not yet rated. This is done by calculating a weighted average of the ratings from similar users, where the weights are their similarity scores. Only animes rated by similar users are considered, and higher similarity scores have a greater influence. The predicted ratings are then sorted in descending order, and the top 10 animes are recommended to the target user, ensuring personalized suggestions based on similar users' preferences.

Following the same process, we implemented KNN, which involves computing the distance between users instead of their correlation.

Table 4.1 presents the recommendations for the same user based on both the correlation and KNN approaches. The results show that the correlation-based approach lacks diversity by predicting optimal ratings for all recommended animes, suggesting a high similarity between the user and the recommendations. On the other hand, the KNN approach provides a variety of ratings reflecting its flexibility in capturing a wider range of user preferences. From this, we conclude that KNN-based methods may offer more personalized recommendations, especially when user tastes are varied.

4.1.2 Item-based collaborative Filtering (IBCF)

IBCF consists of studying the similarity between items to recommend them (such as animes) to users. In this work, we use it to recommend animes that are similar to those users have enjoyed previously

UBCF with correlation		UBCF with KNN	
Anime name	Predicted rating	Anime name	Predicted rating
Hunter x Hunter	10	Ansatsu Kyoushitsu (TV)	8.2
Kimi no Na wa	10	Mirai Nikki (TV)	8.2
Asu no Yoichi!	10	Bleach	8.0
Baccano!	10	Steins;Gate	7.6
Baccano! Specials	10	Re:Zero kara Hajimeru Isekai Seikatsu	7.2

Table 4.1: Top 5 anime recommendations for the same user using UBCF correlation and UBCF KNN

or the content they have watched the most. Similar to UBCF, we have implemented both an IBCF that uses KNN to compute the similarity (cosine) between the anime, as well as the person correlation. Additionally, we used two different kinds of item-based matrices: the pivot matrix from the previous section and a sparse matrix. The sparse matrix is created by mapping user and anime IDs to index positions and then constructing a sparse matrix with ratings as values. This allows for efficient storage of the user-item interaction data.

The KNN algorithm uses the item-user matrix (pivot or sparse) to find similar animes. For the sparse matrix, it maps the given anime ID to its index, extracts its rating vector, and then finds the top k similar animes using the cosine similarity metric. The similar animes are returned by mapping the indices back to the original anime IDs.

To compute the Pearson correlation between two animes using a sparse matrix representation, we extract their rating vectors (columns) and calculate their mean-centered values. Then, the Pearson correlation coefficient is computed by taking the dot product of the mean-centered ratings and normalizing by the product of their magnitudes. Based on that, we compare the target anime with all other animes in the matrix and compute their Pearson correlation to get the top k most similar based on it.

To compare the two approaches, we chose the Pokemon anime to find its 5 top similar animes. The results represented in table 4.2 and table 4.3 show that the pivot matrix gives recommendations that are closely related to the Pokemon series, making it a good choice for fans who want to stick to similar shows. On the other hand, the sparse matrix suggests more varied options, which might interest users looking to discover something different. Each method works well depending on whether the user prefers familiar titles or wants to explore new ones.

IBCF with Pearson correlation	
Pivot matrix	Sparse matrix
Cosmic Break	Boku no Chikyuu wo Mamotte
Circuit Angel: Ketsui no Starting Grid	Wonderful Days
Hashire! Shiroi Ookami	Koutetsu Tenshi Kurumi 2
Cinnamon the Movie	Shichinin no Nana
Puzzle & Dragons CM	Initial D Second Stage

Table 4.2: Top 5 anime recommendations for the same user using IBCF Pearson correlation

IBCF with KNN	
Pivot matrix	Sparse matrix
Pokemon Advanced Generation	Yami no Shihosha Judge
Pokemon: Mewtwo no Gyakushuu	Sakura Taisen: Le Nouveau Paris
Pokemon: Maboroshi no Pokemon Lugia Bakutan	Yumeyume
Pokemon Diamond & Pearl	Hikaru no Go Special
Pokemon: Kesshoutou no Teiou Entei	Juubee-chan: Lovely Gantai no Himitsu

Table 4.3: Top 5 anime recommendations for the same user using IBCF KNN

4.2 Model Based Collaborative Filtering

4.2.1 Matrix Factorization

Matrix factorization is a commonly used model-based collaborative filtering technique that captures latent features of both users and items by decomposing the user-item interaction matrix into lower-dimensional matrices. In this work, we use Singular Value Decomposition (SVD), which is implemented in many Python libraries, with two approaches. The first approach involves using the user-item pivot matrix, where columns represent items (in this case, anime) and rows represent users. Missing ratings are initially filled with zeros, and then the SVD decomposition is performed on this matrix, resulting in lower-dimensional matrices that are multiplied to reconstruct the predicted user-item ratings matrix. To predict new animes for a user, we rely on this reconstructed matrix, which contains the predicted ratings. For recommending new items to a user, the function recommend_for_user retrieves the predicted ratings for the user, sorts the items by these ratings in descending order, filters out the items that the user has already rated, and returns the top n items that the user has not yet interacted with, thus providing personalized recommendations.

The second approach consists of sparsing the pivot matrix since it contains many missing entries. The sparse matrix maps the non-zero ratings from the pivot table into a compact format, storing only the values, their row indices (representing users), and column indices (representing items). Missing values, originally represented as NaN in the pivot table, are handled by filling them with zeros during preprocessing, which the sparse matrix format inherently ignores, focusing only on meaningful interactions for computation. This matrix is then decomposed using SVD, following the same logic as for the pivot matrix to generate anime recommendations for a specific user.

Model	RMSE	MAE
SVD with pivot matrix	7.9090	7.7520
SVD with sparse matrix	**5.0658**	**4.5084**

Table 4.4: Evalution of SVD with pivot and sparse matrices

To compare the two approaches, we generated the top 5 recommendations for the same user, resulting in significantly different outcomes (Table 4.5). This prompted us to evaluate the two methods using the RMSE and MAE metrics. As shown in the table 4.4 the sparse matrix approach outperformed the pivot matrix approach, yielding better results. This can be attributed to the fact that the sparse matrix ignores missing values, while the pivot matrix requires filling them. Filling missing values with zeros caused the model to interpret them as if the user had watched the anime and rated it zero, which was not accurate.

SVD with pivot matrix	SVD with sparse matrix
Aldnoah.Zero 2nd Season	Fullmetal Alchemist
Suzumiya Haruhi no Yuuutsu	Hunter x Hunter (2011)
Zero no Tsukaima: Futatsuki no Kishi	One Punch Man
Sayonara Zetsubou Sensei	Pokemon: Mewtwo no Gyakushuu
Majo no Takkyuubin	Dragon Ball

Table 4.5: Top 5 anime recommendations for the user with the index 0

5 - Contributions

Task	**Contributors**
Collaborative Filtering Recommendation System	Aafaf RIDOUAN
Content-Based Recommendation System	Ismail CHAKRANE
Exploratory Data Analysis	Both contributors
Report Writing	Individual sections written by respective contributors; joint sections collaboratively authored

Table 5.1: Member Contributions

6 - Development Environment

In the `README.md` file of the provided code project, the instructions to rebuild the environment are clearly explained.