FusionAnime: Hybrid Approach for Tailored Recommendations

Venkata Harsha Vardhan Gangla (vgangala@ucsd.edu), Shreya Reddy Pakala (spakala@ucsd.eduu),
Harin Raja Radha Krishna (hradhakrishnan@ucsd.edu), Aman Parikh (amparikh@ucsd.edu)
Universtiy Of California San Diego

1 INTRODUCTION

Anime have become an increasingly popular form of entertainment in recent years. With so many new anime titles being released each year, users often find it difficult to choose which ones to watch. The aim of the project is to build an anime recommendation system using a hybrid recommendation model .i.e using both content and collaborative filtering methods to provide better recommendations to the user. This model is also converted to a web application with which the users can interact and see the recommendation engine in practice.

Github: https://github.com/vgangala10/DSC-148-Final-project-

2 NECESSITY OF HYBRID FILTERING

Content-Based Filtering: The model does not need any data about other users, since the recommendations are specific to this user. This makes it easier to scale to a large number of users. The model can capture the specific interests of a user and can recommend niche items that very few other users are interested in.

Challenges: The model can only make recommendations based on the existing interests of the user. In other words, the model has limited ability to expand on the users' existing interests The data need to be specifically engineered in order to gain accurate recommendations.

Collaborative Filtering: The first challenge it addresses over content-based filtering is that we don't require specifically engineered data. The model recommends items on basis of user-item interaction and can help users discover new interests on basis of the interests of similar users.

Challenges: The main challenge that is faced here is the "Cold Start problem" which is caused due to data sparsity. This means that the model cannot be useful without some significant data and thus cannot make recommendations in the case of new systems.

Therefore, to combat the challenges of both approaches, we have developed a hybrid recommendation system to address the challenges of both the above-discussed models and provide better recommendations.

3 DATASET

3.1 Data Collection

To build a hybrid anime recommendation we require user data and anime data. Myanimelist API is used to extract the data. For this, we obtained the client id and token from myanimelist API. The retrieved data has been stored in useranimelist.csv and anime.csv.

useranimelist.csv: User data has been extracted from API which contains the details of the user's activity on MyAnimeList website. **Features**: anime_id, username, score (rating given by the user), status (completed, plan_to_watch, etc...), is_rewatching. This data set comprises 23 million ratings of about 11000 anime given by approximately 60000 users.

anime.csv: 11299 anime data have been extracted that contains all the details of the anime. **Features**: anime_id, title, genres, main_picture (images), synopsis, num_episodes, start_season, num_list_users (number of users who watched that anime), medium (TV, movie, OVA, etc...), studio.

3.2 Exploratory Data Analysis

We explored most of the data in both files and analyzed many plots for feature selection and the rows that are to be dropped for better predictions. Here, we discussed some of the plots that are worth mentioning.

Initially, we analyzed the user data and then the anime data.

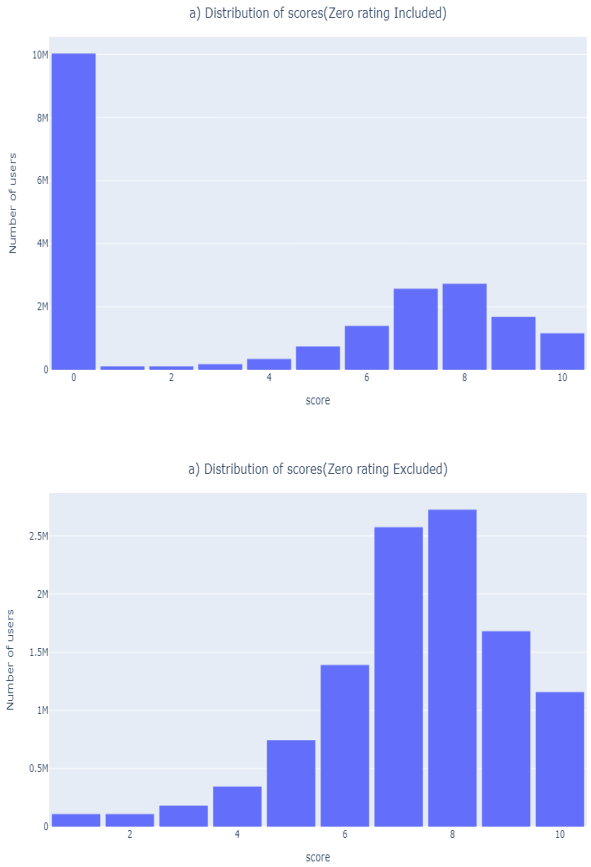

Figure 1: Distribution of scores with zero rating (a) Included (b) Excluded

Firstly, we plotted the histogram for all the ratings of the anime that all the users have rated. From *Figure 1(a)* we can observe

Venkata Harsha Vardhan Gangla (vgangala@ucsd.edu), Shreya Reddy Pakala (spakala@ucsd.eduu),
Harin Raja Radha Krishna (hradhakrishnan@ucsd.edu), Aman Parikh (amparikh@ucsd.edu)

that most of the anime ratings fall under zero but, there is no zero rating when we searched MyAnimeList website. Here, a zero rating indicates that the user hasn't rated. Excluding zero rating, the remaining ratings can be seen as normally distributed in *Figure 1(b)* with a mean somewhere between 7 and 8.

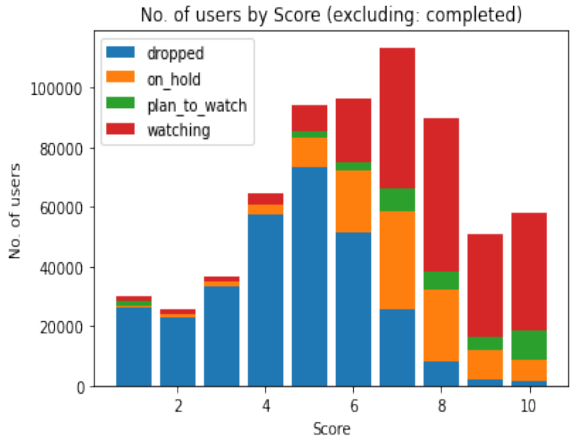

Figure 2: Status vs Scores

Figure 2 shows the stacked bar plot of ratings and the number of users rated, where stacking is done over the status. The users who have completed watching particular anime have been removed as we could not differentiate the number of users having other statuses when stacked over. Here, we found an interesting pattern. The number of users who have dropped watching the anime has rated very low to that anime and this is clearly depicted in the plot. Most of the users who have rated low i.e. from 1 to 6 are the ones who dropped watching the anime. Moreover, it is seen that the users who are yet to watch the anime and have rated tend to rate higher and this is because probably the user is biased towards the anime or the user knows the story is good based on the manga (manga are comics or graphic novels originating from Japan).

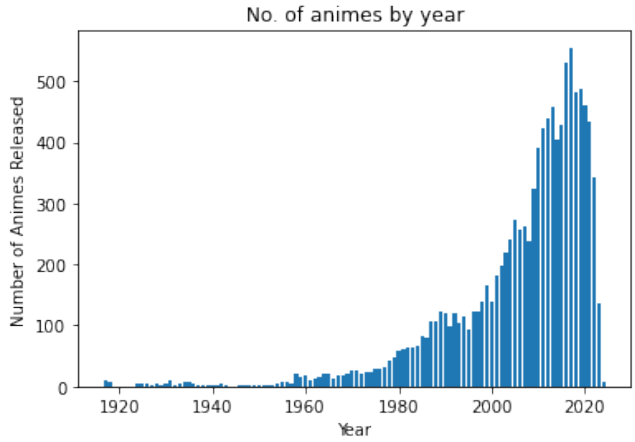

Figure 3: Number of anime released in each year since 1915

Figure 3 shows that the number of anime released has exponentially increased as the years progressed and this really says that people have invested more time in anime and can also infer that the popularity of anime has increased exponentially.

The pie chart (*Figure 4*) depicts the number of anime segregated by their source material. From the chart, we can infer that 4129 anime have original content and 2324 of them are manga adaptations. All the other sources contribute less than a quarter part of all the anime.

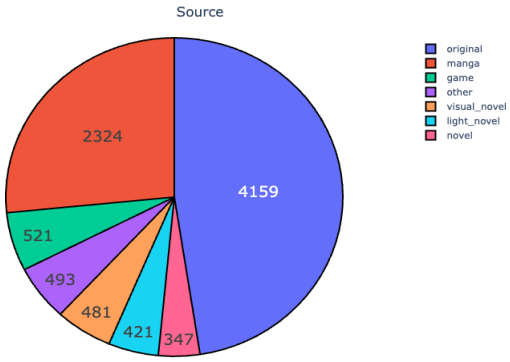

Figure 4: Sources of the anime

The histogram(*figure 5*) shows us the medium of the anime through which it can be accessed. Almost 3500 anime are TV based, which tells us that the animes that are broadcasted on TV are more popular than those broadcasted as movies, OVAs, etc... types. The medium through which an anime is broadcasted can influence the audience it appeals to, as viewers may prefer to watch anime that is aired through a medium they are familiar with.

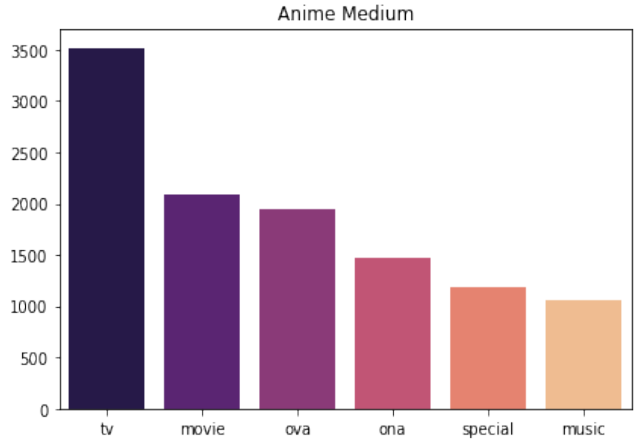

Figure 5: Frequency of anime according to their medium

4 TF-IDF VECTOR OF TEXTUAL DATA

TF-IDF is considered to be one of the best metrics to determine how significant a term in a series or a corpus is to a text. It is a weighting system that assigns a weight to each word in a document based on its term frequency (TF) and the reciprocal document frequency. The words with higher scores of weight are deemed to be more significant.

Term Frequency (TF): In document d, the term frequency represents the number of instances of a given word w.

$$tf(w,d) = count(w,d)/\ N(w,d)$$

where count(w,d) is count of w in d, N(w,d) is number of words in d

Inverse Document Frequency (IDF): IDF is used to measure how relevant a particular term is in a document. To calculate this, we first calculate the Document Frequency (DF) which is the count of occurrence of a word in document d. This is given as :

$$idf(w) = N/\ df(w)$$

where N is the number of documents containing the word w.

Thus combining these, TF-IDF is given as follows:

$$\boxed{tf - idf(w,d) = tf(w,d) * idf(w)}$$

Here, we have used TF-TDF on the 'synopsis' column in our data, to make content-based recommendations on the basis of the content of anime.

5 COSINE SIMILARITY

Cosine similarity is a metric used to measure the similarity of two vectors. Specifically, it measures the similarity in the direction or orientation of the vectors ignoring differences in their magnitude or scale. Both vectors need to be part of the same inner product space, meaning they must produce a scalar through inner product multiplication. The similarity of two vectors is measured by the cosine of the angle between them.

Mathematically,

$$\vec{A} \cdot \vec{B} = \|\vec{A}\|\|\vec{B}\| \cos(\theta)$$

$$\cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\|\|\vec{B}\|}$$

$$\cos(\theta) = \frac{\sum_{i=1}^{n} \vec{A}_i \cdot \vec{B}_i}{\sqrt{\sum_{i=1}^{n} \vec{A}_i^{\,2}} \sqrt{\sum_{i=1}^{n} \vec{B}_i^{\,2}}}$$

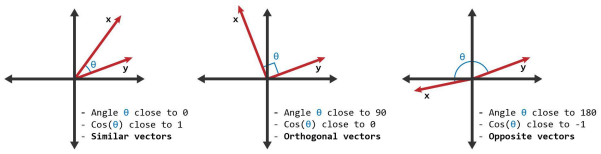

Figure 6: Pictorial representation of cosine similarity

6 COLLABORATIVE FILTERING

6.1 Predictive task

Given the ratings of the anime the users watched, we should recommend anime to the user that would better represent their interest by finding the anime that they might tend to rate the highest based on their previous interactions. This is a collaborative recommendation system.

6.2 Evaluation

Root Mean square Error (RMSE) is calculated based on the original unit of measurement. RMSE is a common evaluation metric. It penalizes larger errors harsher than small errors. It is more sensitive to predictions that are further from the center and can be calculated efficiently. Therefore, in the following tasks, we will evaluate the model performance based on RMSE.

The user data is split into 80:20 train test data to evaluate the scores on the unseen dataset. The model is trained to optimize the parameters using the data in the training dataset. The actual performance is dependent on the average RMSE score applying the model to a similar set of testing data.

6.3 Baseline

We are using three models as baselines: K-NN, Average Ratings, and average ratings with user and anime biases(the rating deviation of the user and anime)

As part of building a recommender system, we have made the assumption that users who have similar preferences or behaviors are likely to have similar tastes in items they may be interested in. To implement this assumption, we have used the K-Nearest Neighbors (KNN) method, which is a widely used technique for measuring similarities between items or users in recommendation systems.

The **surprise** package is a popular python library used for building recommendation systems, and it includes several built-in algorithms for computing similarities between users, including KNN. The KNN algorithm in surprise calculates the similarity between users based on their shared ratings of items. The algorithm identifies the k-nearest neighbors of a user, which are the k users who have rated the most similar set of items to that user.

Result: RMSE for the model is 2.9. All hyperparameters for the model are kept by default values.

The second baseline we propose is the average rating for all users which is also called the global bias. The average rating for our training dataset is 7.28. With this prediction, the test dataset has an RMSE of 1.89

The final baseline we propose is the average rating of the users combined with the user and anime biases. The user and anime biases are |U| and |I| dimension vectors that are initialized to zero initially and further optimized using a cost and derivative function. The cost function is minimized using scipy.optimize package. After optimizing the model, we get an RMSE of 1.71 for the training dataset and an RMSE of 1.71 for the test dataset.

Baseline Result: Among the 3 baselines discussed above, the final model which uses global bias, user bias, and anime biases to predict the ratings did a better job. So with this model as our baseline, we will perform further improvements to reduce the RMSE and give better recommendations.

6.4 Model

Initial Setting.

Dataset: useritemrating

Features: user_id, anime_id, and ratings

We are trying to predict the ratings and give recommendations based on the Latent Factor method. We are trying to optimize the RMSE score of the prediction on the training dataset and test them on the test dataset. All operations that decrease RMSE scores below 2.9 will be considered a significant improvement in performance.

6.4.1 *Latent Factor Model.*

Latent Factor models, also known as Matrix Factorization models,

are feature extractor models used to capture meaningful latent connections between users and items inferred from rating patterns. The latent factor model is a popular collaborative filtering technique used in recommendation systems for predicting user preferences for new items based on their past ratings.

To learn the latent factors, the model uses a matrix factorization approach where the user-item rating matrix is decomposed into two lower-dimensional matrices: one representing user factors and the other representing item factors. The dot product of these matrices generates a prediction matrix that estimates the ratings for all users and items in the dataset.

The model is trained by a loss function that minimizes the difference between the predicted ratings and the actual ratings in the training set using gradient descent. Regularization terms are also added to prevent overfitting and ensure that the latent factors are meaningful and interpretable. The Parameters in the model are described as follows:

$r_{u,i}$ = Rating of user u on anime i
$\hat{r}_{u,i}$ = Predicted rating of user u on anime i
μ = Overall mean anime rating
b_u = Rating deviation of user u
b_i = Rating deviation of anime i
q_i = Latent factor of anime i
p_u = Latent factor of user u
λ_1 = Regularization term for user and item bias
λ_3 = Regularization term for user and item latent factors

To predict a user's rating for a new anime title, the model computes the dot product of the user's latent factors and the title's latent factors. The resulting value represents the predicted rating for the user and the title. The model is as follows:

$$\hat{r}_{u,i} = \mu + b_u + b_i + q_i^T p_u + \lambda_1 \sum_{u=1}^{U} ||b_u||^2 + \lambda_1 \sum_{i=1}^{I} ||b_i||^2$$

$$+\lambda_2 \sum_{u=1}^{U} \sum_{k=1}^{K} ||p_u||^2 \lambda_2 \sum_{i=1}^{I} \sum_{k=1}^{K} ||q_i||^2$$

6.4.2 *Feature Engineering*.

We found that a significant portion of the values in our rating feature within the useranimelist contained 0 values. However, on the website, the rating system starts at 1. Therefore, a value of 0 denotes that a user has viewed the anime but has not rated it. Since we cannot infer the rating pattern of users based on these 0 values, we have removed the user-item rating entries with those values.

Moreover, not all of the information about the anime that users rated in the useranimelist were extracted from the anime data. Therefore, we removed any user-rated anime for which we did not have complete anime details. This was necessary to ensure that we could display all the anime details on our application. Additionally, we have removed any anime with less than 30 user ratings and users who have rated fewer than 50 anime to obtain a more accurate representation of users and their rating patterns for anime.

6.4.3 *Hyper parameter tuning*.

To learn the latent factors, the model uses a matrix factorization approach where the user-item rating matrix is decomposed into two lower-dimensional matrices: one representing user factors and the other representing item factors. The dimension of this matrix is trained for values 8, 16, 32, and 64.

Also, the regularization terms are also added to prevent overfitting and ensure that the latent factors are meaningful and interpretable. We use two lambda values of different strengths to regularize the weights of the model: one for the user and item biases and the other for the latent factors for the user and item.

6.4.4 *Final Optimization Result*.

We used the scipy.optimize package to optimize the user, item features, and user, item latent factors. The dimension of the latent features that fit the test dataset was 32. We also tuned the values of lambda1 and lambda2 to get the least RMSE on the test dataset. The best combination of lambda1 and lambda2 was for values 0.001 and 0.0015 respectively. The RMSE of the tuned model over the test dataset was 1.6, which was almost similar to the train RMSE of 1.58.

The model performed much better than the baseline model of K-Nearest neighbors which had an RMSE of 2.9. where Therefore, we choose to propose the Latent Factor Model as our final model in recommending animes to the users.

```
Enter your username:TerFig
```

	title	genres	synopsis
976	Death Note	['Psychological', 'Shounen', 'Supernatural', '...	Brutal murders, petty thefts, and senseless vi...
1597	Code Geass: Hangyaku no Lelouch R2	['Action', 'Award Winning', 'Drama', 'Mecha', ...	One year has passed since the Black Rebellion,...
4826	Shingeki no Kyojin Season 2	['Action', 'Drama', 'Gore', 'Military', 'Shoun...	For centuries, humanity has been hunted by gia...
7072	Violet Evergarden Movie	['Award Winning', 'Drama', 'Fantasy']	Several years have passed since the end of The...
10654	Shingeki no Kyojin: The Final Season - Kankets...	['Action', 'Drama', 'Gore', 'Military', 'Shoun...	The conclusion to Shingeki no Kyojin.

Figure 7: Recommendations based on collaborative filtering

From *Figure 7* we can see the anime recommendations based on the user. Once the name of the user is provided the model uses the factorized matrix to predict the ranting and recommends the top five anime based on the predicted ratings.

7 CONTENT-BASED FILTERING

A content-based recommender system tries to guess the features or behavior of a user given the item's features, which they react positively towards. The recommendation given by content-based filtering highly depends on the features it is trained on. Generally, many content-based systems provide recommendations on either the basis of the content of the item or the item features like genres. We performed content-based filtering by combining both approaches and providing recommendations accordingly.

7.1 Data Cleaning

The data used for performing content-based filtering is the Anime.csv dataset. The original size of the data is : (11299, 8) The following shows an overview of the data(only the columns under consideration for the model are displayed).

	title	synopsis	processed_synopsis	genres	source	medium
0	Cowboy Bebop	Crime is timeless. By the year 2071, humanity ...	Crime timeless . By year 2071 , humanity expan...	['Action', 'Adult Cast', 'Award Winning', 'Sci-...	original	tv
1	Cowboy Bebop: Tengoku no Tobira	Another day, another bounty—such is the life o...	Another day , another bounty—such life often u...	['Action', 'Adult Cast', 'Sci-Fi', 'Space']	original	movie
2	Trigun	Vash the Stampede is the man with a $$60,000,0...	Vash Stampede man 60,000,000,000 bounty he...	['Action', 'Adult Cast', 'Adventure', 'Sci-Fi'...	manga	tv
3	Witch Hunter Robin	Robin Sena is a powerful craft user drafted in...	Robin Sena powerful craft user drafted STNJ—a ...	['Action', 'Detective', 'Drama', 'Mystery', 'S...	original	tv
4	Bouken Ou Beet	It is the dark century and the people are suff...	It dark century people suffering rule devil , ...	['Adventure', 'Fantasy', 'Shounen', 'Supernatu...	manga	tv

Figure 8: Overview of the data

From the data, all the anime without any synopsis have been removed. The shape of the data is : (9442, 9). Also, the null values in the 'source' column have been replaced by 'other'.

7.2 Feature Engineering

Processed Synopsis : The synopsis column has also been pre-processed, removing the stopwords, tokenizing, and lemmatizing the data. This pre-processed synopsis is stored as the 'processed_synopsis' back into the data and is then used for the model.

TF-IDF Vectorization : The processed synopsis column in the data provides a basic overview of the anime. TF-IDF transformation has been applied to this column to get the weights of different words in the text. The values are then normalized and are further used to discover what animes might be similar based on their synopsis.

One Hot Encoding : We then used One Hot Encoding to get the relevant features with respect to individual anime. We considered the following columns for the purposes of one hot encoding:

(1) Genres: Genres/Genre thee Anime belongs to.
(2) Source: Source Anime belongs
(3) NSFW: Anime's Category(white/gray).

Since each anime could belong to more than one genre, the genre values were given as a list. We explored the genre column so as to get the individual genre names and then performed one hot encoding on the column.

7.3 Predictive task

Given the anime user selected from the collaborative filtering model and using the features (genres, source, nsfw) and content (processed synopsis) of the anime from the data, we recommend the most similar anime to the user that would better represent their interest.

7.4 Model

Cosine Similarity : Cosine Similarity is then applied to both the TF-IDF matrix and one hot-encoded matrix to find similarity scores between anime. Since either taking only TF-IDF or one hot encoded matrix would only base the recommendations on either one of the approaches, we decided to average out the similarity scores of both to provide both with equal importance. Thus both TF-IDF and one hot encoding similarity scores are averaged to get the final cosine similarity scores keeping anime titles as an index.

Output : The final Cosine similarity scores are then provided to a custom function that outputs the list of the top 10 anime with respect to the highest similarity scores across anime.

	title	genres	synopsis
7150	One Piece Movie 14: Stampede	['Action', 'Adventure', 'Fantasy', 'Shounen']	Monkey D. Luffy and the Straw Hats arrive aboa...
3684	One Piece Film: Z	['Action', 'Adventure', 'Fantasy', 'Shounen']	The Straw Hat Pirates enter the rough seas of ...
4766	One Piece 3D2Y: Ace no shi wo Koete! Luffy Nak...	['Adventure', 'Fantasy', 'Shounen']	After suffering great personal loss during the...
435	One Piece Movie 06: Omatsuri Danshaku to Himit...	['Action', 'Adventure', 'Drama', 'Fantasy', 'S...	"If you are a pirate among pirates among pirat...
5309	One Piece: Episode of Sabo - 3 Kyoudai no Kizu...	['Action', 'Adventure', 'Fantasy', 'Shounen']	The special will revisit the childhood past of...
1923	One Piece Film: Strong World	['Action', 'Adventure', 'Fantasy', 'Shounen']	Upon hearing news that islands in East Blue ar...
430	One Piece Movie 01	['Action', 'Adventure', 'Fantasy', 'Shounen']	Many years ago, Woonan, a legendary pirate, pl...
10509	One Piece: Barto no Himitsu no Heya!	['Comedy']	Bartolomeo and a few mysterious guests recap t...
10471	One Piece Film: Red	['Action', 'Adventure', 'Award Winning', 'Come...	As a child, Uta—the Red Hair Pirates' ex-music...
433	One Piece Movie 04: Dead End no Bouken	['Action', 'Adventure', 'Fantasy', 'Shounen']	Luffy and crew arrive at the harbour of Anabar...

Figure 9: Top 10 recommended anime based on content-based model

8 LITERATURE

Matrix Factorization Techniques for Recommender Systems, by Yehuda Koren and Robert Bell, and Chris Volinsky. The authors review various matrix factorization algorithms, including classical methods such as Singular Value Decomposition (SVD) and Alternating Least Squares (ALS), as well as more recent methods such as Non-negative Matrix Factorization (NMF) and Deep Matrix Factorization (DMF). They also discuss the use of regularization and side information in matrix factorization-based recommendation models. While the paper provides a comprehensive survey of matrix factorization techniques for recommender systems, they have not touched mostly on the discussion of hybrid methods. The paper focuses primarily on matrix factorization techniques and does not provide a comprehensive review of hybrid methods that combine matrix factorization with other recommendation algorithms that might tend to better cater to the needs of the users. In our model, we implemented Collaborative based recommendations to users, and based on their interaction with the recommended animes we suggest similar animes based on the content.

Anime - Content Collaborative - KNN, by Hernan Valdivieso. [link]
In Anime - Content Collaborative - KNN project, Hernan has used content-based filtering to suggest anime that are similar. He used several features of the anime from the anime dataset, and found the movies that are similar to each other based on K- the nearest neighbors method and suggested the user those movies. He does the same for the user ratings, where he finds users who have similar ratings based on their previous ratings. He finds the nearest users with similar rating patterns and suggests the movie the nearest users have liked. The project assumes that the ratings of the users will be the same as similar users and doesn't take into consideration that there might be some pattern or correlation between the user and the type of anime they like. In our model, we try to overcome this shortcoming by using latent features in the model that implicitly tries to capture the relationship between user preferences and anime features.

Venkata Harsha Vardhan Gangla (vgangala@ucsd.edu), Shreya Reddy Pakala (spakala@ucsd.eduu),
Harin Raja Radha Krishna (hradhakrishnan@ucsd.edu), Aman Parikh (amparikh@ucsd.edu)

Content based Filtering sorted by weighted average, by Raksh. [link]
In "Content based Filtering sorted by weighted average", Raksh proposes an anime recommendation model based on the weighted average features. He first uses KNN-based collaborative filtering, to recommend anime and later compares this approach with a weighted average recommendations model based on total reviews of the anime and anime average rating. The weighted recommendation model tries to give more emphasis to anime that are most popular and have a higher rating and most views, than solely trusting the recommendations from the model. The model does a better job of predicting but doesn't consider the impact of the synopsis, medium and other features of the anime information that might also have a greater say while giving recommendations based on content. In our model, we give emphasis on other features in the anime dataset by giving different weights to the synopsis and the other features that have been one hot encoded in our model that might have an impact on the final recommendation. Also, we have built this system on top of collaborative filtering instead of only relying on one model for a recommendation.

9 RESULTS

Our collaborative model outperformed the baseline by giving an RMSE score of 1.6. The model provides a list of recommendations for a particular user and we stacked a content-based model on top of the recommendations given by the collaborative model. We believe this will help better represent the interests of the user.
We developed a web application with **Sreamlit** which utilizes the described models to fetch anime recommendations. There are three components to this application. The first component is the HOME page (*Figure 9*) that displays the top 10 anime based on the number of users who watched them, which can be watched within a weekend, and the ones belonging to the genres action or adventure.

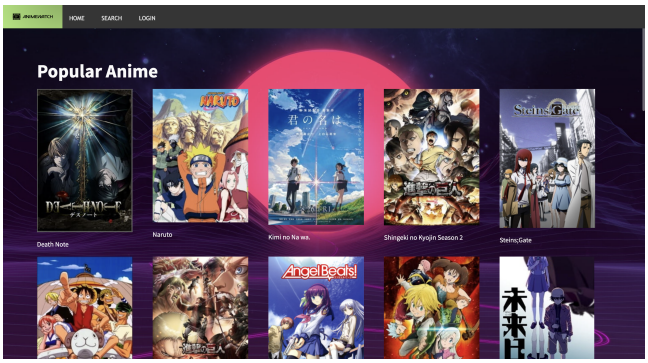

Figure 10: Home page of the web application

The second component is the LOGIN page (*Figure 10*), where the users are expected to give their username and password to see personalized recommendations given by the collaborative filtering approach. We authenticate the username against the usernames present in our data that is fetched from MyAnimeList API. Once login is successful, the user can see the top 10 recommendations along with their genre, source, and summary. *Figure 11* shows the recommendations for username 'BongBong73'.

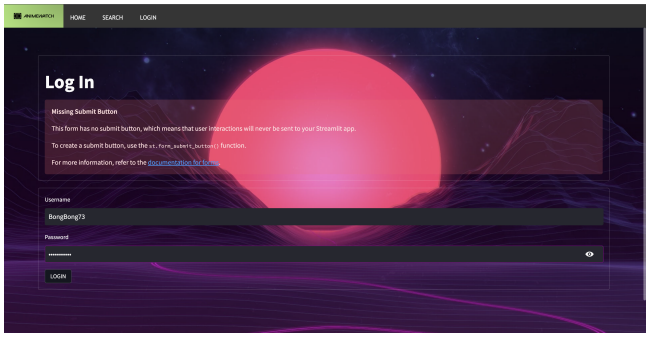

Figure 11: Login page of the web application

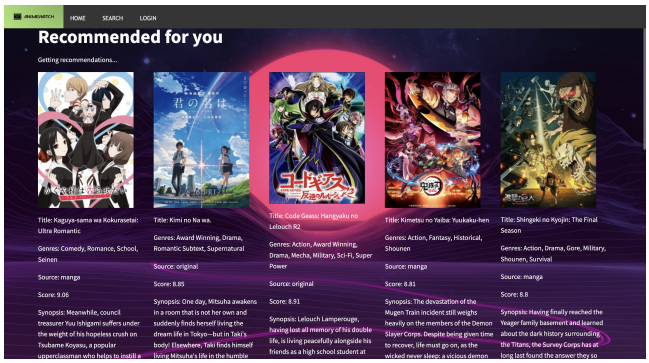

Figure 12: Recommendations given by the collaborative filtering approach

The final component is the SEARCH page, which takes the anime title from the user as input and uses the content-based approach to recommend the top 10 similar titles. The search field is an auto-select drop box. For every anime recommended, we display its poster, title, genre, source, and summary. Let us say that the user searched for the anime Kimetsu no Yaiba: Yuukaku-hen based on the above recommendations. *Figure 12* shows the anime recommended by the content-based model.

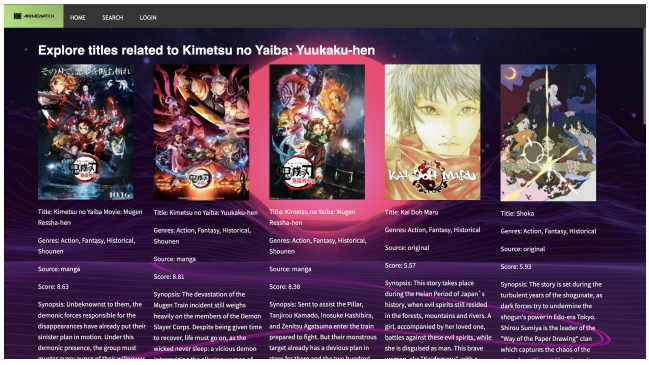

Figure 13: Recommendations given by the content-based approach

10 MAJOR TAKEAWAYS

- While using collaborative methods to recommend anime, considering the interaction between the user and anime through latent factors performs a better role in recommending anime as it gives the lowest RMSE score compared to the other models.

- Factorizing the user-item rating matrix is a better way to represent the rating data in a low-dimensional space. And they do a better job in recommending and predicting the ratings.
- The train and test accuracies for the latent factor method were nearly the same, despite the large test dataset. This depicts that the model does a better job of fitting the data without overfitting.
- When we performed the latent factor method with a relatively high number of latent dimensions say 64, it took several hours to run the model. But still, the performance of the model on the test data was lower than with 32, indicating that with higher than optimal k, the model tends to overfit the data.
- In content-based filtering, features - synopsis, genres, source, NSFW, and the medium were highly beneficial in giving better recommendations. Common features like synopsis, genres, production companies, and release time are beneficial in predicting movie ratings.
- A combined approach of averaging cosine similarity applied separately to the synopsis(engineered through TF-IDF) feature and all the other features tends to give a better recommendation than applying cosine similarity to the TF-IDF to all the features combined.
- Both approaches used different features in recommending the animes based on the requirement. The latent factor model used just 3 feature columns to recommend, while the content-based model used several features from TF-IDF to recommend. Both the models seem to fit the data very well and generalize well to the larger crowd.

11 FUTURE WORKS

- A weighted average of Content-based and collaborative models can be used for better recommendations. Based on the animes recommended using collaborative filtering, we can recommend similar animes to those on the same page using content-based filtering.
- Cold start problem can be addressed for the collaborative filtering model. Users not in our database won't be able to login, but still can get content-based recommendations based on the animes they search on the page.
- More advanced machine learning algorithms, such as deep learning or reinforcement learning can be integrated into the hybrid models.
- Methods for modeling the evolution of user preferences and item popularity over time to make more accurate recommendations can be studied and implemented.

12 REFERENCES

[1] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8
[2] Chen,Tianqi, et al. "Feature-based matrix factorization." arXiv preprint arXiv:1109.2271 (2011)
[3] Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," in Computer, vol. 42, no. 8, pp. 30-37, Aug. 2009, doi: 10.1109/MC.2009.263.
[4] Content based Filtering sorted by weighted average, by Raksh Available at: https://www.kaggle.com/code/raksh710/content-based-filtering-sorted-by-weighted-average/notebook
[5] Anime - Content Collaborative - KNN, by Hernan Valdivieso Available at: https://www.kaggle.com/code/hernan4444/anime-content-collaborative-knn
[6] Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model, Yehuda Koren
[7] A Toscherd, M. Jahrer, R. Bell, The BigChaos Solution to the Netflix Grand Prize
[8] CooperUnion, Anime Recommendations Database—Kaggle Available at: https://www.kaggle.com/CooperUnion/anime-recommendations-database