Data Mining and Advanced Statistical Methods in Business Applications

Final Project

Movie Recommendation System

Group 2

1. Trần Hải Nam: 11219279
2. Nguyễn Hà Phương: 11219284
3. Vũ Trọng Mạnh: 11213752
4. Lê Hoàng Anh Đức: 11219268
5. Nguyễn Minh Tú: 11216040

Table of Contents

I. Introduction

1.1. Problem definition

A recommender system is essentially a system/model/algorithm intended to suggest relevant items to the users. It can be movies, music, and many more. Generally speaking, recommendations will be extremely needed when it comes to the relationship between users and service providers or buyers and e-commerce. In the end, decent recommendations will be a win-win solution that benefits both parties as the users are more loyal because they got what they wanted and the service providers gained more profit. You may think, how impactful those recommendations are? In fact, it's massive.

1.2. Motivation

According to McKinsey, recommendations play a crucial role in:

- **40%** of app installs on Google Play
- **60%** of watch time on Youtube
- **35%** of purchases on Amazon
- **75%** of movies watched on Netflix

So, it's a good idea to learn how to build one, right?

1.3. Problem setting

1.3.1. Dataset

This dataset contains metadata for all 45,000 movies listed in the Full MovieLens Dataset. The dataset consists of movies released on or before July 2017. Data points include cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages.

This dataset also has files containing 26 million ratings from 270,000 users for all 45,000 movies. Ratings are on a scale of 1-5 and have been obtained from the official GroupLens website.

This dataset consists of the following files:

movies_metadata.csv: The main Movies Metadata file. Contains information on 45,000 movies featured in the Full MovieLens dataset. Features include posters, backdrops, budget, revenue, release dates, languages, production countries and companies.
keywords.csv: Contains the movie plot keywords for our MovieLens movies. Available in the form of a stringified JSON Object.
credits.csv: Consists of Cast and Crew Information for all our movies. Available in the form of a stringified JSON Object.

links.csv: The file that contains the TMDB and IMDB IDs of all the movies featured in the Full MovieLens dataset.
links_small.csv: Contains the TMDB and IMDB IDs of a small subset of 9,000 movies of the Full Dataset.
ratings_small.csv: The subset of 100,000 ratings from 700 users on 9,000 movies.

1.3.2. Goal

The primary objective of this project is to provide users with personalized movie recommendations based on their preferences, using a combination of different filtering techniques.

II. Literature Review

Movie recommendation systems play a crucial role in assisting users in discovering movies that align with their preferences. This literature review focuses on the hybrid method employed in such systems, which combines content-based and collaborative filtering approaches to enhance recommendation accuracy and coverage.

2.1. Content-Based Filtering

Content-based filtering suggests movies to users based on the similarity between movie attributes and user preferences. It analyzes features such as genre, cast, director, and plot to provide personalized recommendations. Previous research by J. Lops, M. de Gemmis, and G. Semeraro (2011) provides insights into the state of the art and trends in content-based recommender systems. The objective of implementing a content-based recommender is to offer personalized movie recommendations by analyzing intrinsic movie features, thereby enhancing user satisfaction and engagement.

2.2. Collaborative Filtering

Collaborative filtering recommends movies by examining user interactions and preferences, assuming that users who have agreed in the past will agree again in the future. Previous research by Y. Koren, R. Bell, and C. Volinsky (2009) discusses matrix factorization techniques, including Singular Value Decomposition (SVD), for collaborative filtering. The objective of implementing collaborative filtering techniques is to harness user interactions to provide personalized movie recommendations, aiming to improve recommendation accuracy and relevance.

2.3. Hybrid Recommendation Approach

The hybrid recommendation approach combines both content-based and collaborative filtering methods to enhance recommendation accuracy and coverage. This approach leverages the strengths of both methods while mitigating their weaknesses. Research by various authors (e.g., Burke, 2002; Adomavicius & Tuzhilin, 2005) discusses the effectiveness of hybrid recommendation systems in providing more accurate and diverse recommendations compared to individual approaches. By combining content-based and collaborative filtering, the hybrid method aims to offer users personalized and relevant movie suggestions, thereby enhancing their movie consumption experience.

The main object of this research is to develop a comprehensive movie recommendation system that integrates content-based and collaborative filtering approaches in a hybrid method. The system aims to provide personalized movie recommendations by analyzing both intrinsic movie features and user interactions. By combining these methods, the paper aims to offer users highly accurate and diverse movie suggestions, ultimately improving their movie consumption experience.

III. EDA

3.1. Distribution of Adult and Non Adult Movies

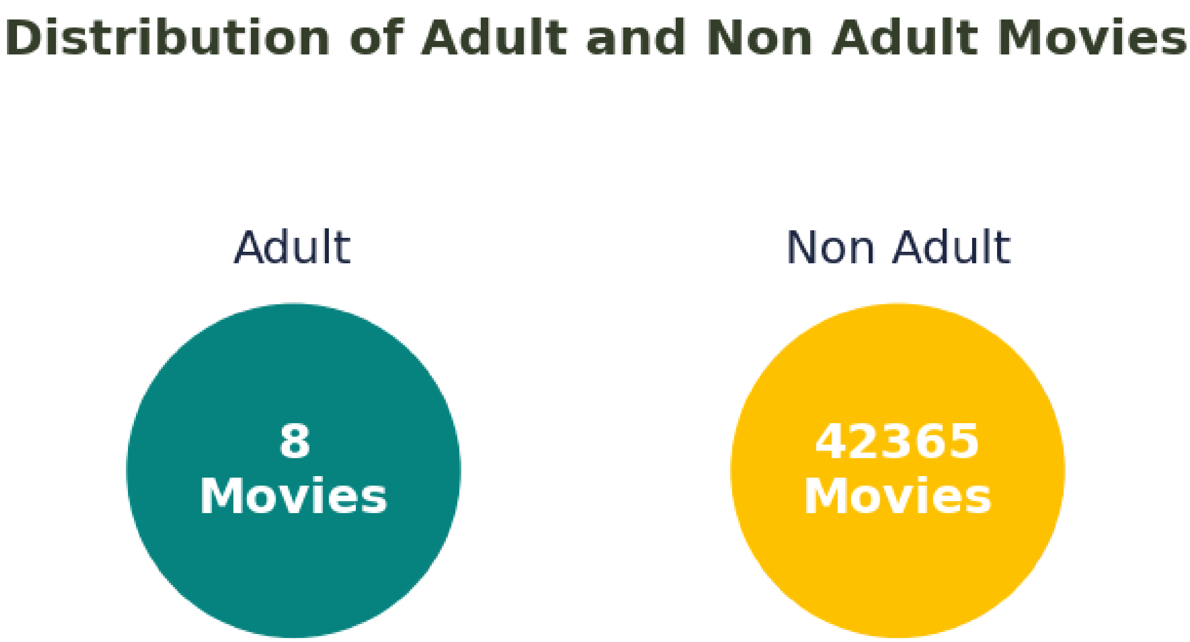


Figure 1. Distribution of Adult and Non-adult Movies

There is a huge difference between those 2 types of movies in terms of presence for this particular dataset (8 adult movies VS 42,365 non-adult movies).

3.2. The Influence of Budget and Revenue on Popularity of Movies

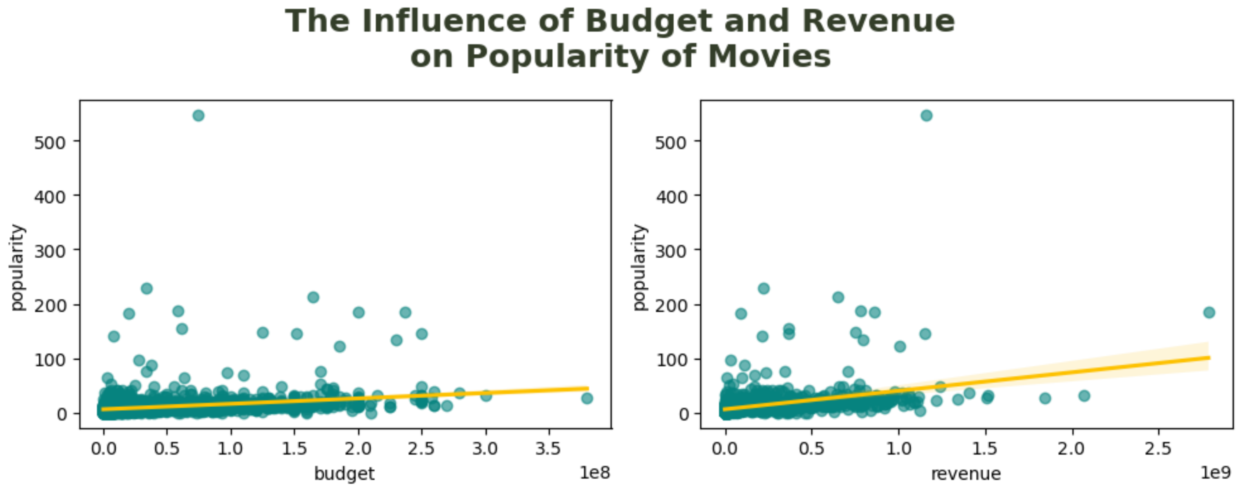


Figure 2. The Influence of Budget and Revenue on Popularity of Movies

Budget and Revenue just slightly influence the popularity of the movies.

3.3. Budget VS Revenue

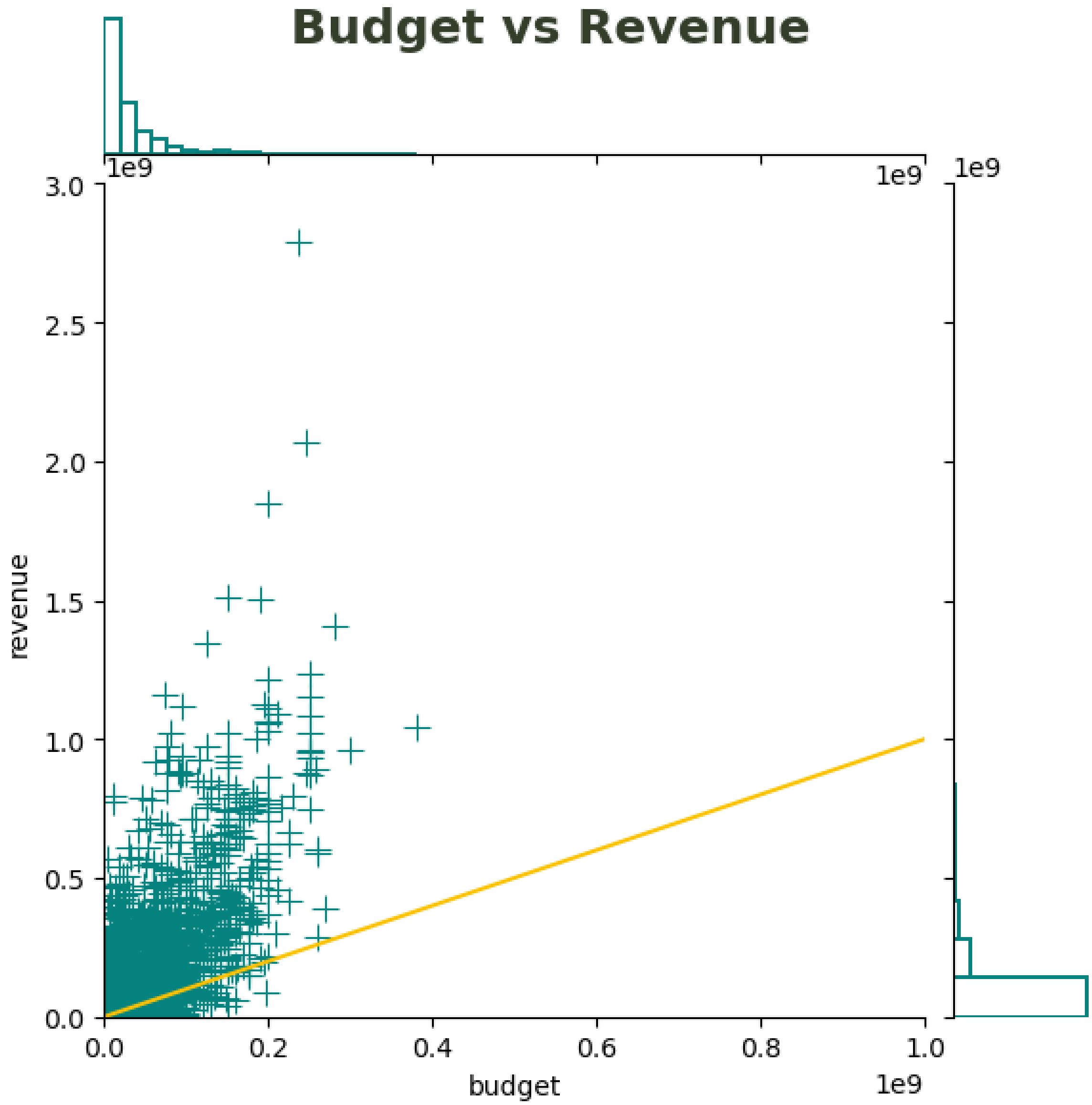


Figure 3. Budget VS Revenue

Most of the movies lay on top of the yellow line, indicating that those movies make a profit.

3.4. The most common word in Movie Overviews

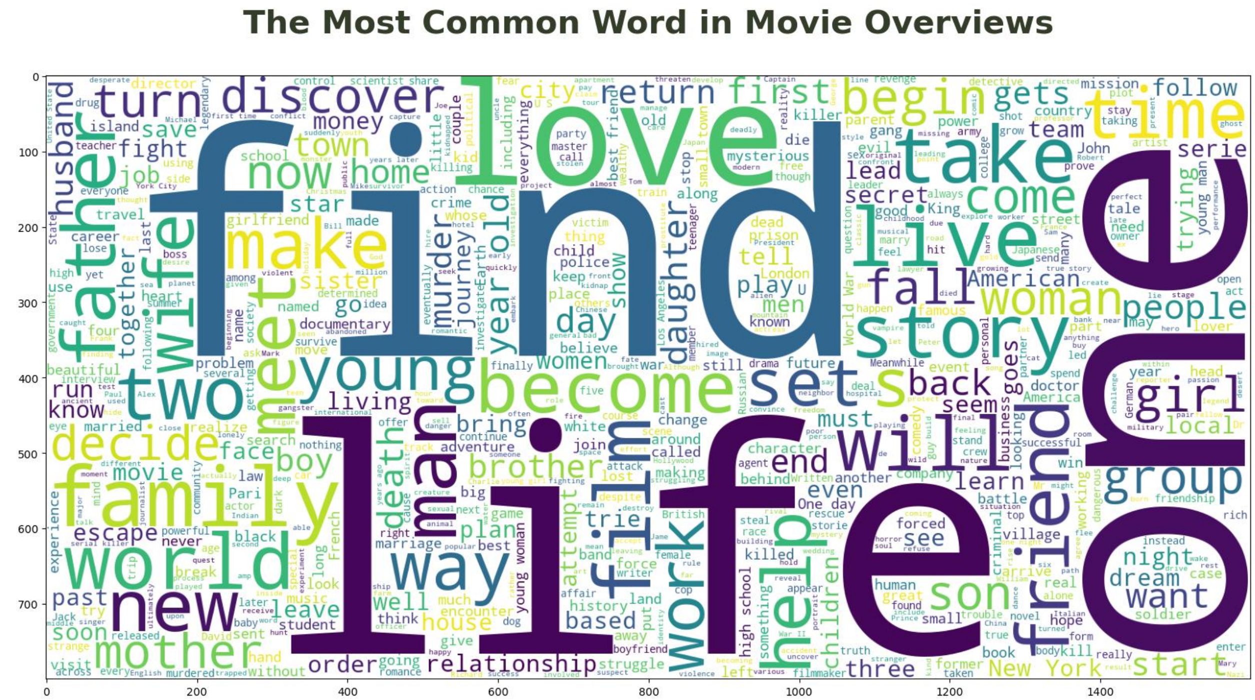


Figure 4. The most common Word in Movie Overviews

Words "life", "one", "find", and "love" apparently appear on many occasions.

3.5. Top 5 genres in Movies

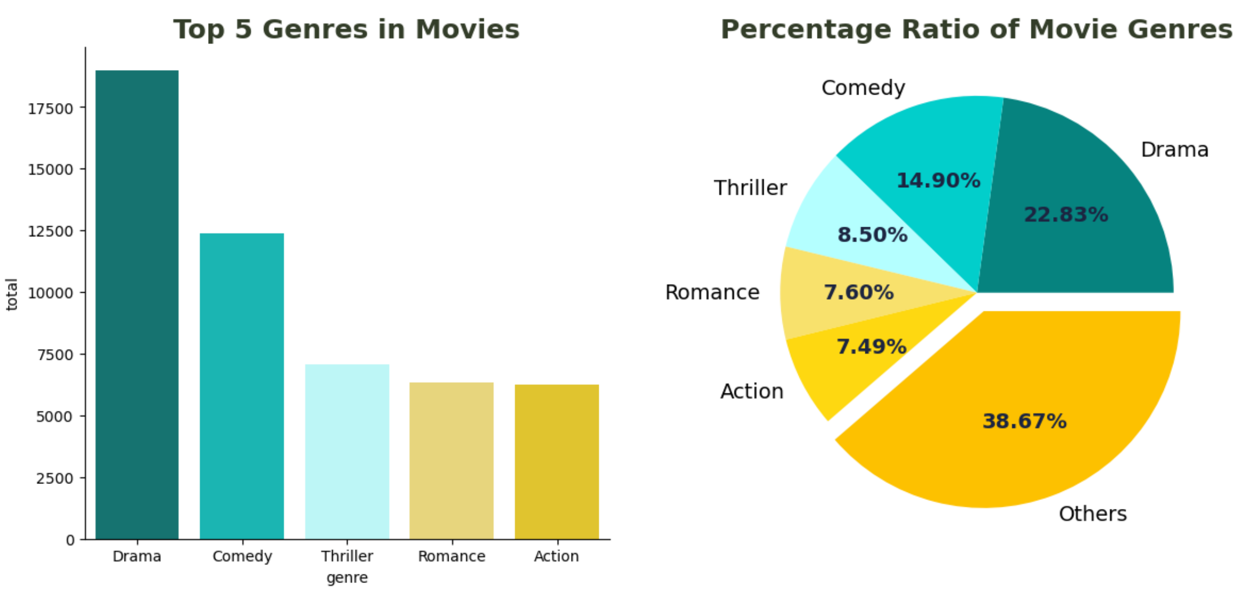


Figure 5. Top 5 genres in Movies

Drama is the most dominant genre with over 18,000 movies. Out of 5 top genres (Drama, Comedy, Thriller genre, Romance, Action), there are still many genres in the dataset. They hold 38.67% of the total genres in the movies.

3.6. Total released movie by Date

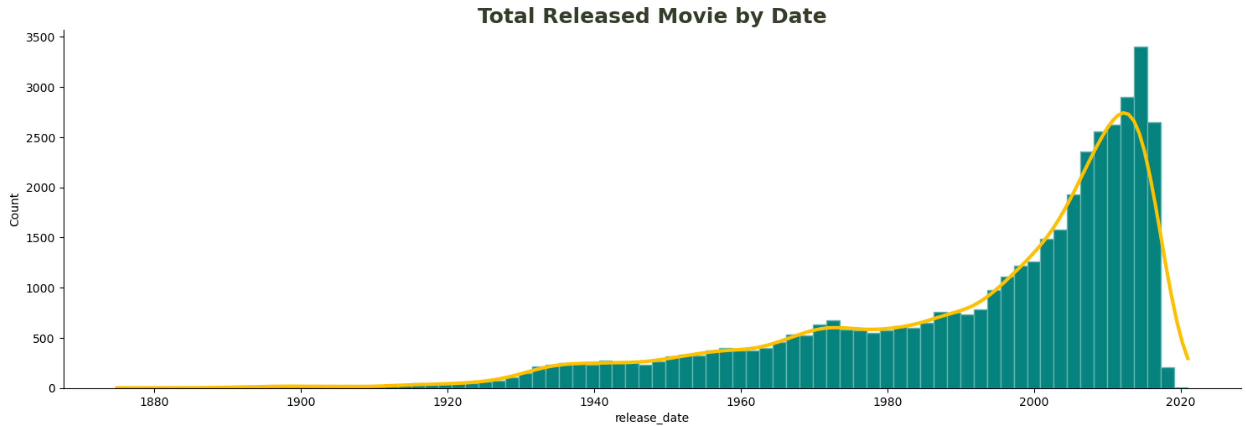


Figure 6. Toral Released Movie by Date

Starting in 1930, the movie industry had grown significantly from 50 years ago. A drop in total released movies around 2020 is because the dataset only contains a few data in those years.

3.7. Top 5 in spoken_language, original_language, actors, crews, production_companies, production_countries in Movie

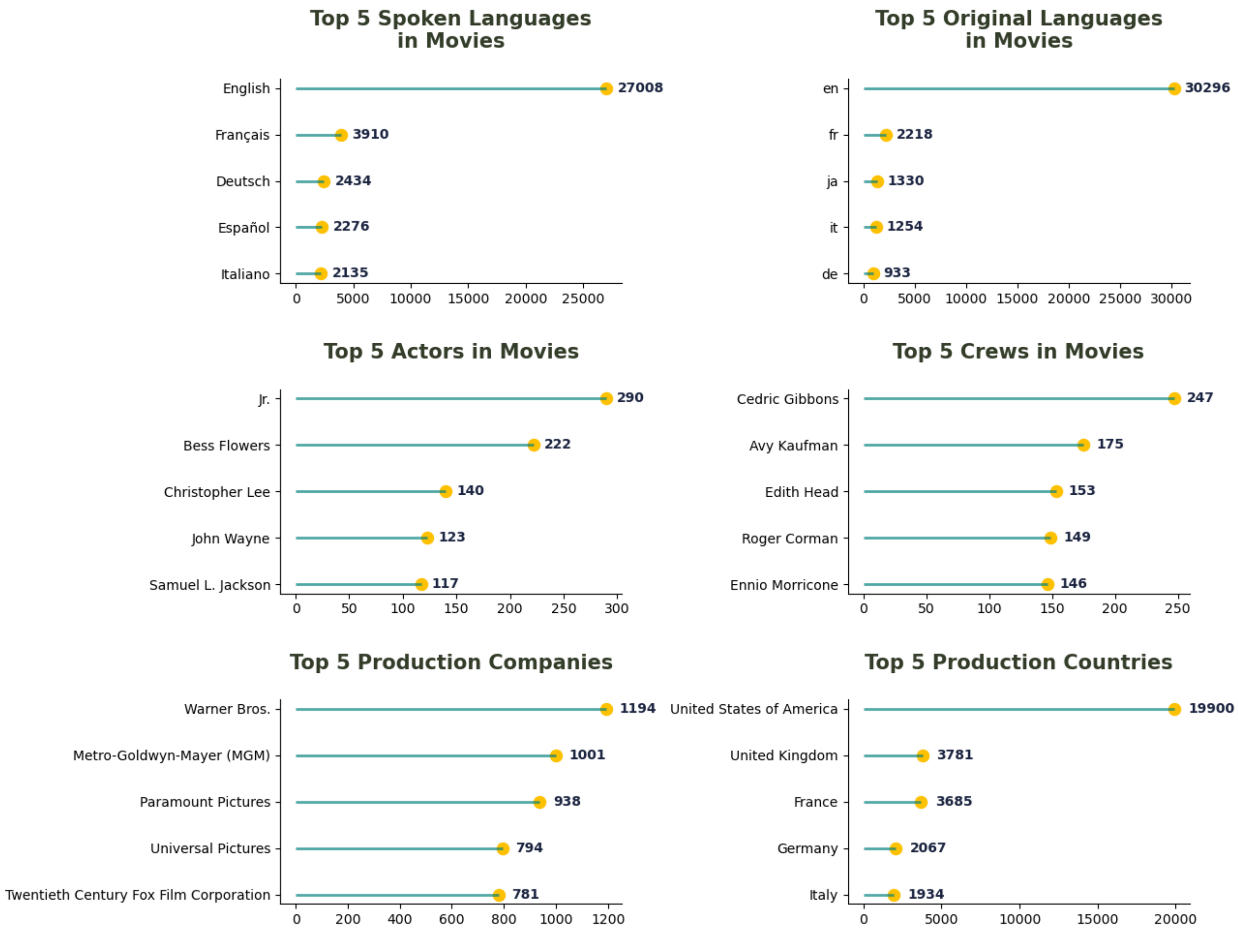


Figure 7. Top 5 in spoken_language, original_language, actors, crews, production_companies, production_countries in Movie

- For this particular dataset, English is on top of the list for the original and spoken language in the movies
- Jr. and Cedric Gibbons are actors and crew involved in the most movies in the list respectively
- Warner Bros. with 1194 movies makes it become top 1 production company on the list
- Many great production companies come from the USA. So, it's not a surprise if the USA has become our top 1 production country.

3.8. The relationship between Rating and Popularity

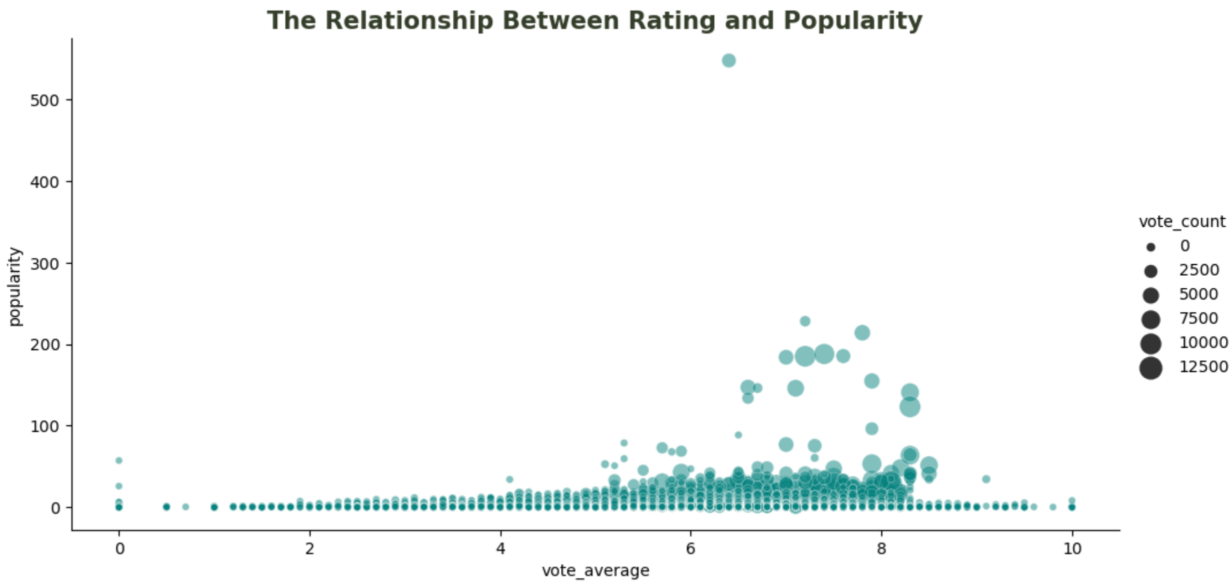


Figure 8. The relationship between Rating and Popularity

A small number of voters basically caused movies that either got a rating of 0 or 10. As the vote count increases, the rating is most likely around 5 to 8.5. Popular movies will get more vote count as shown by the above plot.

3.9. Data distribution across top 5 genres

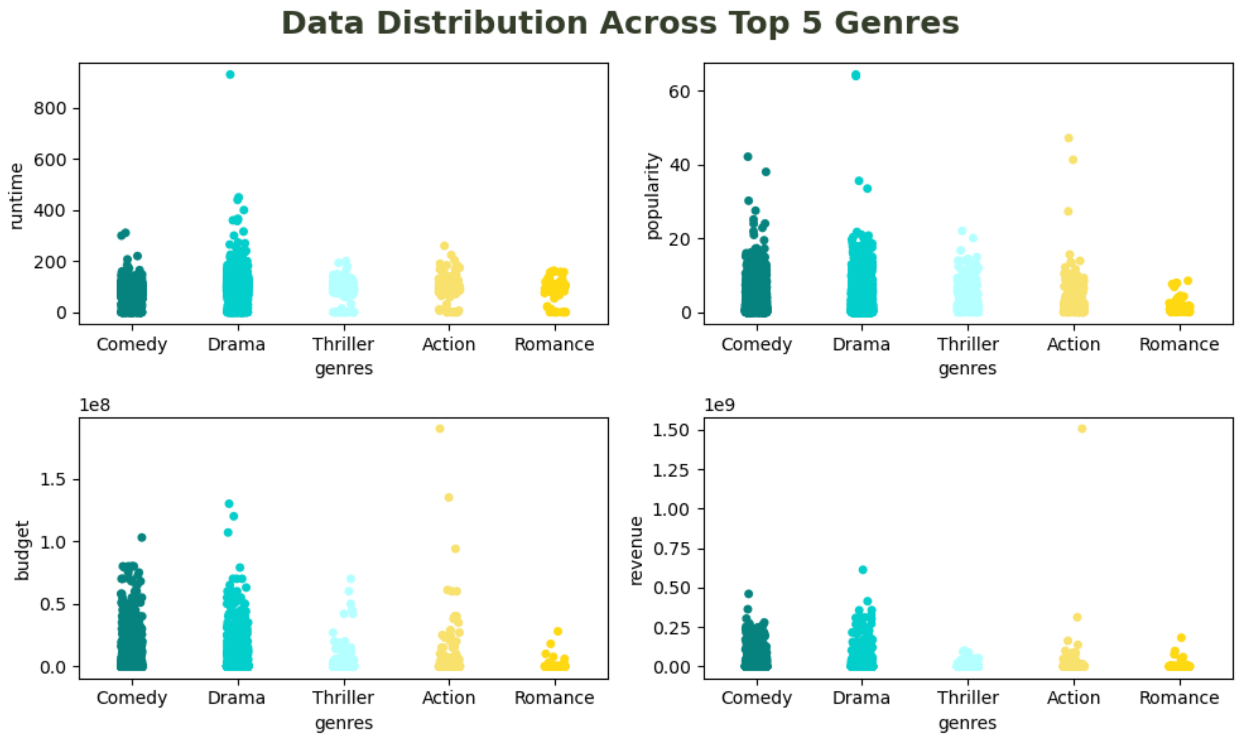


Figure 9. Data distribution across top 5 genres

- The movie genre that has the longest runtime is drama.
- The least popular genre in the top 5 is romance.
- Action movies spent more money than the rest of the movies
- One of the action movies made a vast profit compared to the others.

3.10. Correlation of Movie Features

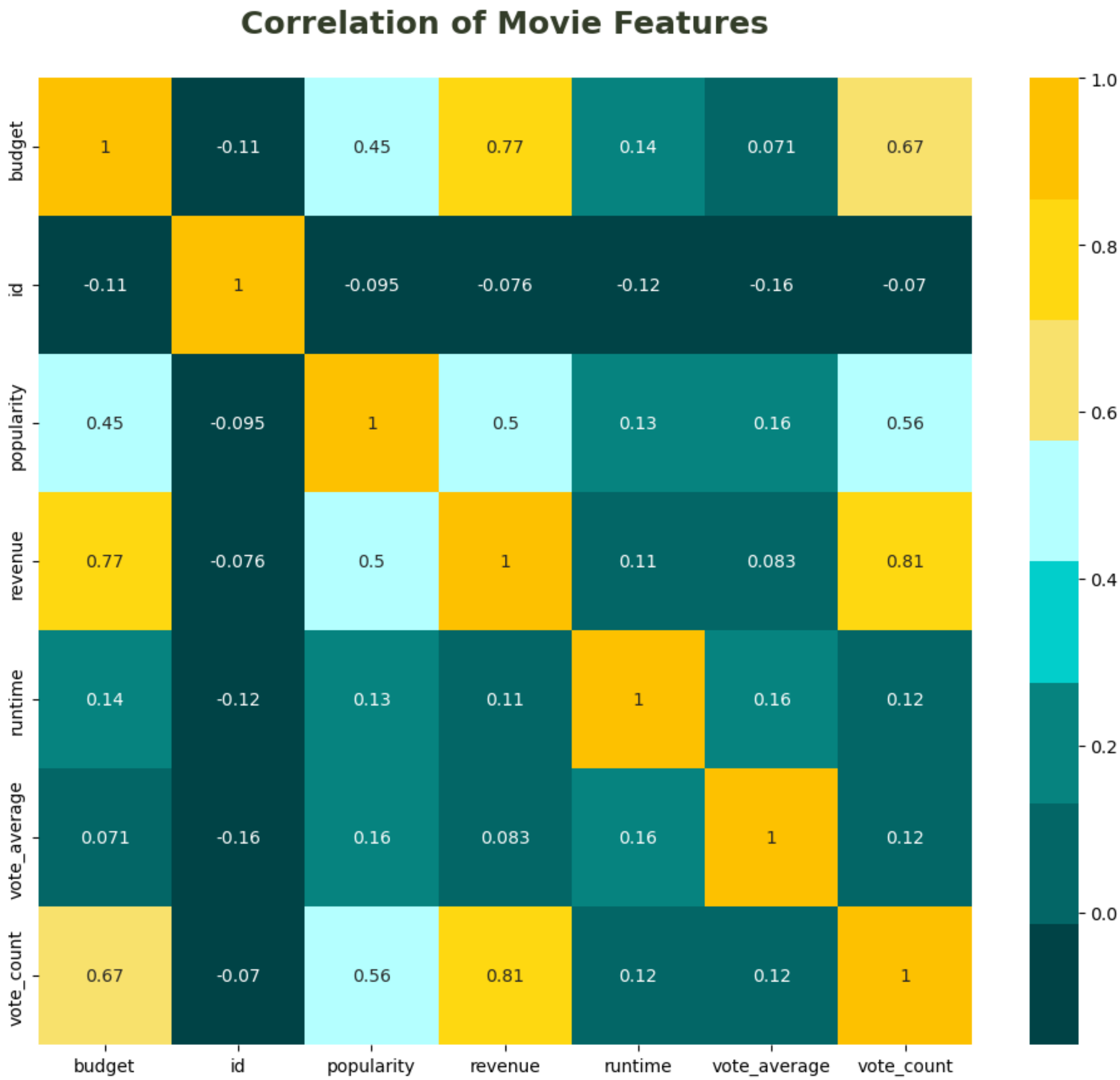


Figure 10. Correlation of Movie Features

Vote count, budget, and popularity are 3 dominant features that will determine the revenue of the movies.

IV. Solution

4.1. Task 1: Content-based Filtering

Content-Based Filtering recommends movies by analyzing the content and features of each movie. This method takes into account user preferences and suggests movies with similar characteristics to those the user has enjoyed in the past.

We have been learning the recommendation system for a while, we may be familiar with the weighted average. The idea behind it is to give a "fair" rating for each movie. In this project, we took it to the next level with the help of bag of words.

In the dataset we had, there are tons of valuable information such as genre, overview,... So we used this information to make our recommendation system more robust. We extracted that information inside bag of words and then combined it with weighted average to get the final similarity for the movies.

$$W = \frac{R\,.\,v + C\,.m}{v + m}$$

Where

- W: Weighted rating
- R (Rating): Average for the movie as a number from 1 to 10 (mean)
- v (Votes): Number of votes for the movie
- m: Minimum votes required to be listed in the Top 250 (currently 25.000)
- C: The mean vote across the whole report (currently 7.0)

original_title	score	similarity	final_score
Toy Story	0.502729	1.000000	0.850819
Toy Story 2	0.460652	0.533157	0.511405
Toy Story 3	0.489255	0.275823	0.339853
Toy Story of Terror!	0.422935	0.299218	0.336333
Small Fry	0.375331	0.271239	0.302467
Hawaiian Vacation	0.389081	0.264853	0.302121
WALL·E	0.508330	0.199791	0.292353
Finding Nemo	0.495991	0.197794	0.287253
Up	0.510838	0.173065	0.274397
A Bug's Life	0.411551	0.204652	0.266722

In this demonstration, movies similar to 'Toy Story' have been identified, showcasing the capability of the Content-Based filtering approach to provide personalized recommendations based on content characteristics.

This method is particularly valuable for suggesting movies that share thematic elements, genres, or content features, delivering a personalized viewing experience aligned with the user's preferences. Besides, content-based filtering provides personalized recommendations based on the specific attributes of movies that a user has enjoyed in the past. This personalization enhances user satisfaction and increases the likelihood of users finding movies they truly enjoy, especially in recommendations for users with unique tastes. There is no Cold Start or Sparsity Problem because Content-based filtering does not need data on other users. Another advantage is that content-based recommendation systems can provide explanations of their recommendations by listing features that caused an item to be recommended. Users can easily understand the rationale behind each recommendation, which fosters trust and enhances user experience.

On the other hand, content-based filtering relies heavily on past user preferences to make recommendations, which may never recommend items outside the user's content profile. Users may miss out on discovering new and unexpected movies they would enjoy but have not been exposed to through their past interactions. Moreover, the accuracy of content-based recommendations depends heavily on the quality and completeness of item attributes used for analysis. The system's ability to make relevant recommendations may be compromised if the attributes used for content analysis are limited or inaccurate.

4.2. Task 2: Collaborative Filtering

4.2.1. Why using Collaborative Filtering?

Collaborative Filtering identifies patterns in user behavior and recommends movies based on the preferences of users with similar tastes. For this, our project utilizes the Surprise library, a Python scikit for building and analyzing recommender systems.

- *Dataset* from Surprise: A module to load a dataset.
- *Reader* from Surprise: A module to define the format of input file.
- *SVD* (Singular Value Decomposition) from Surprise: A collaboraive filtering algorithm based on matrix factorization.
- *Model_selection* from Surprise: A module providing tools for model selection and evaluation.

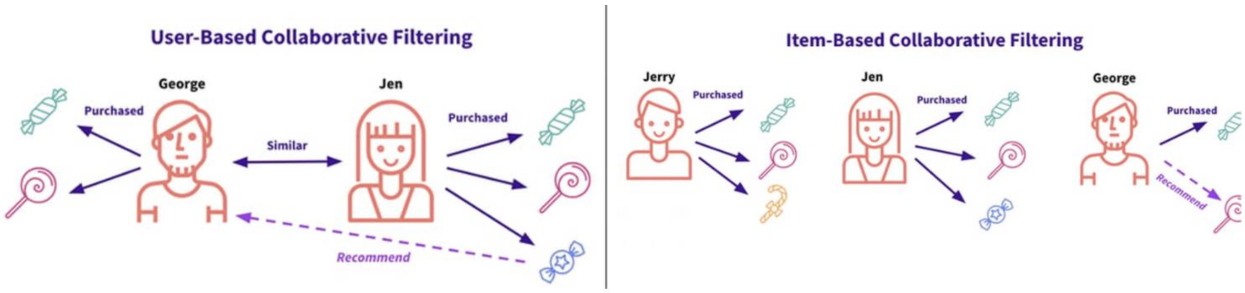
Indeed, Collaborative Filtering excels in capturing latent features and uncovering user preferences based on their interactions. It offers a valuable approach to provide personalized recommendations, particularly when content-based information is limited.

4.2.2. User-based and Item-based Filtering

There are 2 main types of Collaborative Filtering (CF) techniques, namely User-based and Item-based Filtering.

User-based Collaborative Filtering makes recommendations by analyzing the preferences of a target user and finding similar users with comparable tastes. For instance, if a user shares similar ratings across movies with another user, it's presumed they have akin interests. Consequently, if the analogous user has watched and enjoyed a movie that the target user hasn’t seen, it would be recommended to them.

On the other hand, Item-based Collaborative Filtering suggests items based on the similarity between items themselves, rather than users. This method recommends items that are akin to those the active user has previously shown preference for. For example, if a user enjoyed a book from the "Lord of the Rings" series, another book from the same series would likely be recommended.



a. User-based CF b. Item-based CF

Both User-based and Item-based Collaborative Filtering rely on the collective behavior of users to generate personalized recommendations. This collective behavior is utilized to establish similarities either between users or between items, thereby enhancing the recommendation process.

4.2.3. Matrix Factorization (SVD and SVD++)

SVD factorizes a rectangular m x n matrix A into three matrices, U, and VT. U is an m x m matrix, is an m x n diagonal matrix and VT is an n x n matrix. U captures information about the rows of A and VT captures information about the columns of A. The columns of U and VT are called the left and right singular vectors and the diagonal values of are called the singular values of A (square roots of the eigenvalues). U and V are orthogonal matrices which says that all columns are orthogonal to the other columns and are unit vectors (also, $UT = U - 1$ and $VT = V - 1$). The SVD takes the following form:

$$A = U . \sum . V^T$$

In recommendation system, the above equation is transform to

$$A = P . Q = (U . \sqrt{\sum}) . (\sqrt{\sum .} V^T)$$

to be interpretable. The prediction for score of user *u* for item *i* might be:

$$\widehat{r_{ui}} = \mu + b_{item} + b_{user} + q_{item}{}^T . p_{user}$$

Where $\sum$ is the global mean, b_{item} is the offset vector of the item, b_{user} is the user offset vector.

The Loss function of SVD++ is defined as:

$$J = \sum(r_{ui} - \widehat{r_{ui}}) + \lambda(||p_{user}||^2 + ||q_{item}||^2 + b_{user}{}^2 + b_{item}{}^2)$$

4.2.4. Neighborhood-based Models (KNN)

K-nearest neighbors (KNN) is a type of supervised learning algorithm used for both regression and classification. KNN tries to predict the correct class for the test data by calculating the distance between the test data and all the training points. The algorithm calculates the probability of the test data belonging to the classes of 'K' training data and the class holds the highest probability will be selected.

In our work, we use K-NN inspired algorithm, to be specific, we use Mean Squared Difference instead of Euclidean Distance. Both of them are measurements of distance but MSD is more suitable to use in collaborative filtering where the values is a better representation of user ratings. we also tried another type of KNN by adding 'mean' parameters to deal with users who always rate too high or too low ratings. and another special thing is we only include neighbors for which the similarity measures is positive.

4.2.4.1 KNN basic

To predict user's rating, the first step is to compute similarities between users. This means that for a given user, the algorithm finds other users who have similar taste as this user, based on the ratings they have given to various items. Then it uses the ratings from those similar users to predict the unknown ratings for the given user.

The predict formula comes as:

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} sim(u,v) * r_{vi}}{\sum_{v \in N_i^k(u)} sim(u,v)}$$

- r_{ui}: This is the predicted rating of user u for item i.
- $N_i^k(u)$: This is the set of k items rated by user u that are most similar to item i. In other words, these are the k nearest neighbors of item i based on the ratings by user u.
- $sim(u,v)$: This is the similarity between user u and user v.

To calculate similarity, we need to compute the Mean Squared Difference between all pairs of users.

$$MSD(u,v) = \frac{1}{|I_{uv}|} * \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2$$

- I_{uv}: This is the number of items rated by both user u and user v.
- r_{ui}: This is the rating of user u for item i.
- r_{vi}: This is the rating of user v for item i.

Then, we measure the similarity.

$$sim(u, v) = \frac{1}{MSD(u, v) + 1}$$

4.2.4.2 KNN with mean

We also tried another type of KNN to deal with customers who tend to always give too high or too low rating.

The only difference in this KNN is we take into account the mean rating of all users.

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in N_i^k(u)} sim(u, v) * (r_{vi} - u_v)}{\sum_{v \in N_i^k(u)} sim(u, v)}$$

Where:

- μ_u: is the mean rating of user u.
- $N_i^k(u)$: is the set of k items most similar to item i that were rated by user u.
- $sim(u, v)$:is the similarity between users u and v.
- r_{vi}: is the rating of item i by user v.
- μ_v: is the mean rating of user v.

From two algorithm above, we have a result table from user-based approach:

	RMSE	MAE	Run time
KNN basic	0.943	0.73	11.32s
KNN with mean	0.918	0.705	12.09s

We see that KNN with 'mean' have a higher score. This could mean that some users tend to rate slightly more extreme than others.

4.2.5. Evaluation

No	Algorithm	RMSE	MAE
1	Baseline Item-based CF	2.5511	2.2585
2	Baseline User-based CF	1.7188	1.3484
3	Baseline SVD	0.9041	0.6974
4	**SVD++**	**0.8986**	**0.6924**
5	KNN Basic (user-based)	0.943	0.73
6	KNN Mean (user-based)	0.918	0.705

Table 1: Comparision of Models Performance in Collaborative Filtering

The Baseline CF algorithms have relatively high RMSE and MAE values. This suggest ted that these algorithms may not be as accurate in their predictions compared to others.

The Baseline SVD shows a significant improvement from the first two, indicates a more accurate performance.

The KNN basic and KNN mean also shows a good performance with pretty low scores. We also see that user-based approach in our work bring a better score, this also apply to our KNN.

The SVD++ algorithm here is an extention of SVD that is specifically designed for recommendation system has the lowest score among all. By generating implicit feedback refers to users's history information, SVD++ give the best personalized recommendation in our work.

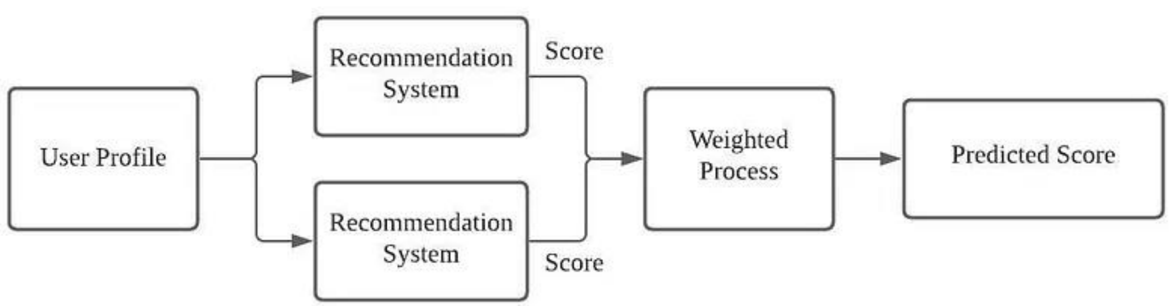
4.3. Task 3: Hybrid Approach

4.3.1. Definition

The hybrid recommendation system is a special type of system that used data of both collaborative data and content-based data simultaneously which helps to suggest a similar or close item to the users. Combining the two above approaches helps to resolve the big problems in more effective cases sometimes.

4.3.2. Types of Hybrid Recommendation System

4.3.2.1. Weighted recommendation system

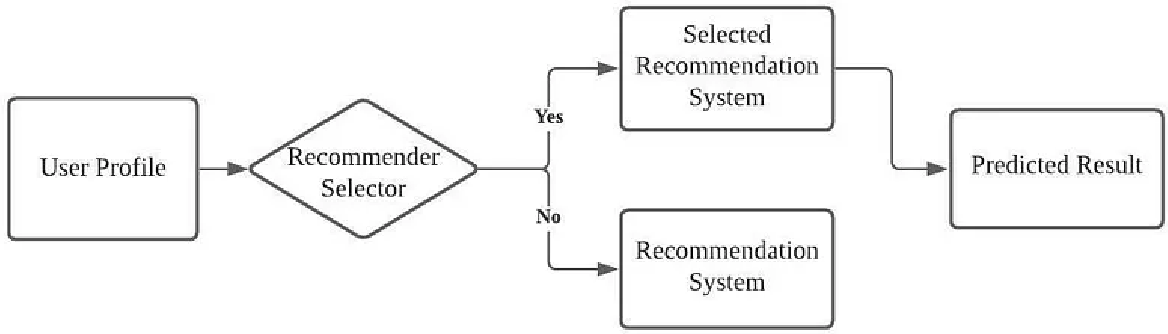


In the weighted recommendation system, we can define a few models that is able to well interpret the dataset. The weighted recommendation system will take the outputs from each of the models and combine the result in static weightings, which the weight does not change across the train and test set.

For example, we can combine a content-based model and a item-item collaborative filtering model, and each takes a weight of 50% toward the final prediction.

The benefit of using the weighted hybrid is that we integrate multiple models to support the dataset on the recommendation process in a linear way.

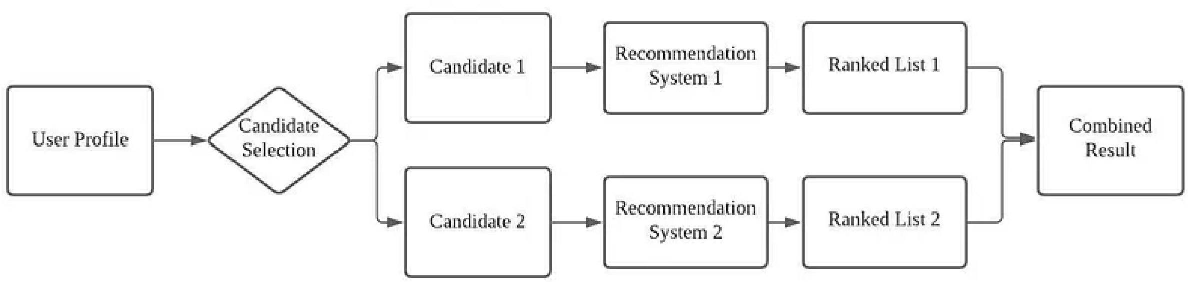
4.3.2.2. Switching



The switching hybrid selects a single recommendation system based on the situation. The model is used to be built for the item-level sensitive dataset, we should set the recommender selector criteria based on the user profile or other features.

The switching hybrid approach introduces an additional layer upon the recommendation model, which select the appropriate model to use. The recommender system is sensitive to the strengths and weakness of the constituent recommendation model.

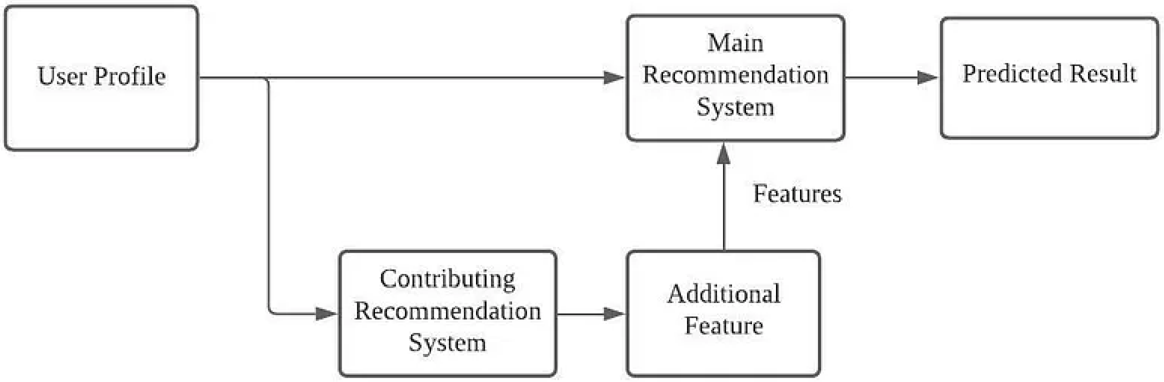
4.3.2.3. Mixed



Mixed hybrid approach first takes the user profile and features to generate different set of candidate datasets. The recommendation system inputs different set of candidate to the recommendation model accordingly, and combine the prediction to produce the result recommendation.

The mixed hybrid recommendation system is able to make large number of recommendations simultaneously, and fit the partial dataset to the appropriate model in order to have better performance.

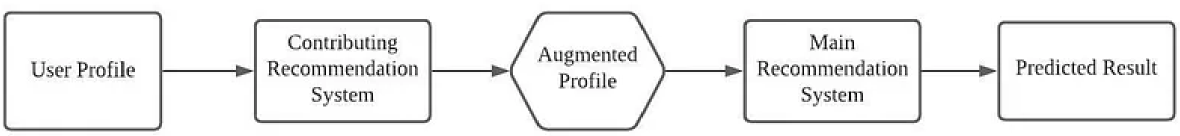
4.3.2.4. Feature Combination



In feature combination hybrid, We add a virtual contributing recommendation model to the system, which works as feature engineering toward the original user profile dataset.

For example, we can inject features of a collaborative recommendation model into an content-based recommendation model. The hybrid model is capable to consider the collaborative data from the sub system with relying on one model exclusively.

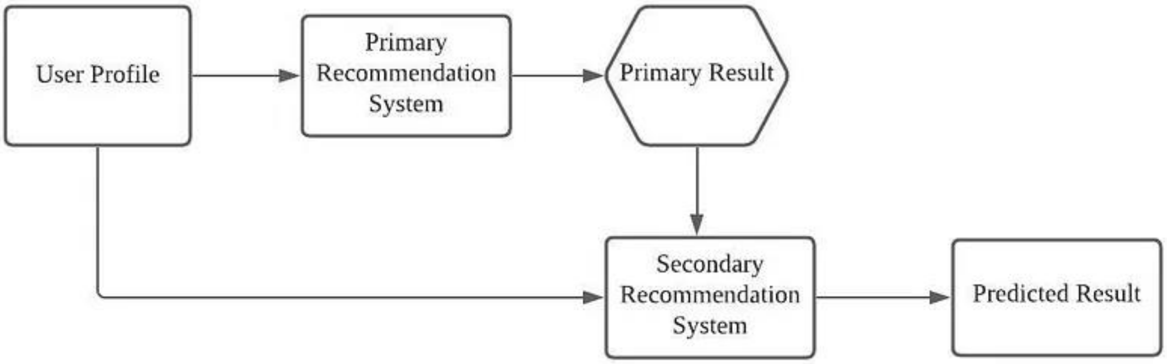
4.3.2.5. Feature Augmentation



A contributing recommendation model is employed to generate a rating or classification of the user/item profile, which is further used in the main recommendation system to produce the final predicted result.

The feature augmentation hybrid is able to improve the performance of the core system without changing the main recommendation model. For example, by using the association rule, we are able to enhance the user profile dataset. With the augmented dataset, the performance of content-based recommendation mode will be improved.

4.3.2.6. Cascade



Cascade hybrid defines a strict hierarchical structure recommendation system, such that the main recommendation system produce the primary result, and we use the secondary model to resolve some minor issues of the primary result, like breaking tie in the scoring.

In practice, most of the dataset are sparse, the secondary recommendation model can be effective against equal scoring issue or missing data issue.

4.3.2.7. Meta-level

Meta-level hybrid is similar to the feature augmentation hybrid, such that the contributing model is providing augmented dataset to the main recommendation model. Different from the feature augmentation hybrid, meta-level replaces the original dataset with a learned model from the contributing model as the input to the main recommendation model.

4.3.3. Hybrid recommendation system implement

The pipeline for this task works as the figure above. The input include the user id and the movie last seen of them. Firstly, we use the content-based filtering to find the top 50 movies similar to the input movie and predict the ratings of this user for 50 movies above by collaborative-filtering and SVD model and making recommendations by top 10 movies that has the highest estimated ratings.

	title	est
2514	Toy Story 2	3.224177
7589	Toy Story 3	3.203545
3817	Monsters, Inc.	3.140483
8555	The Lego Movie	3.024008
1875	A Bug's Life	3.020554
7133	A Matter of Loaf and Death	2.985126
4317	The Looney, Looney, Looney Bugs Bunny Movie	2.947804
2743	Creature Comforts	2.881752
7360	Cloudy with a Chance of Meatballs	2.879039
8479	Toy Story of Terror!	2.813607

UserId = 1, Movie = 'Toy Story'

	title	est
8555	The Lego Movie	3.529343
7589	Toy Story 3	3.455460
7360	Cloudy with a Chance of Meatballs	3.371939
3817	Monsters, Inc.	3.284562
6502	Monster House	3.244078
6464	Cars	3.244075
4317	The Looney, Looney, Looney Bugs Bunny Movie	3.198106
7218	Kung Fu Panda: Secrets of the Furious Five	3.159527
1875	A Bug's Life	3.109210
2514	Toy Story 2	3.074789

UserId = 500, Movie = 'Toy Story'

The recommended movies by hybrid model are better significantly personalize than the only content-based filtering. Moreover, hybrid recommendation system not only improve making recommendation system, but also solve the cold start problem that the collaborative-filtering often suffers from.

V. Conclusion and Discussion

5.1. Conclusion

In conclusion, the aim of this paper is to review all the basic methods in the recommendation system, setting the foundation for further work. Our exploration of recommendation approaches has revealed the distinct advantages and limitations of both content based and collaborative filtering methods. Especially in the latter with 2 popular methods. Neighborhood models are most effective at detecting very localized relationships but ignore the vast majority of ratings. Meanwhile, Latent factor models are generally effective at estimating overall structure that relate simultaneously to most or all items but poorly detecting strong relations among small sets of closely related movies.

5.2. Future work

Future research could explore several avenues to enhance recommendation systems based on our findings. A good recommender system is often contextual, considering the situation at which a recommendation should be made, e.g., what product is the user currently browsing on, or time information such as weekday vs weekend. Thus, other contexts like time, location should be extracted into the model as well.Secondly, the more sparse data is, the worse performance of matrix factorization method, so the application of Deep Learning is essential.

VI. References

1. k-NN inspired algorithms by Surprise

2. Sambandam Jayalakshmi, Narayanan Ganesh,Robert Čep, and Janakiraman Senthil Murugan; 2022. Movie Recommender Systems: Concepts, Methods, Challenges, and Future Directions
3. Nisha Sharma, Mala Dutta; 2020. Movie Recommendation Systems: A Brief Overview
4. Ke Ma; 2016. Content-based Recommender System for Movie Website
5. Srs Reddy, Sravanu Nalluri, Subramanyam Kunisetti, S Ashok; 2018. Content-Based Movie Recommendation System Using Genre Correlation
6. Narendra Kumar Rao, Nagendra Panini Challa, S Sreenivasa Chakravarthi, R Ranjana; 2022. Movie Recommendation System using Machine Learning
7. Chaurasia Neha Sunilkumar; 2022. A review of movie recommendation system: Limitations, Survey and Challenges
8. Prof. Snehal Thorave, Abhay Sharma, Sahil Nair, Tanmay Shinde, Mayur Tipras; 2022. SURVEY ON MOVIE RECOMMENDATION SYSTEM USING ML