

Media Synergy: A Unified Recommendation System

Ashwini Ramesh Benni
Master's in Data Science
dept. Mathematical Sciences
Hoboken, USA
abenni@stevens.edu

Gahana Nagaraja
Master's in Data Science
dept. Mathematical Sciences
Hoboken, USA
gnagaraj1@stevens.edu

Vaishnavi Rajendra Dhotargavi
Master's in Data Science
dept. Mathematical Sciences
Hoboken, USA
vdhotarg@stevens.edu

Abstract— In the era of information abundance facilitated by rapid technological advancements and widespread internet access, recommendation systems have emerged as crucial tools to address information overload. In the digital age, where individuals have unparalleled access to diverse media content, ranging from music and movies to books, the challenges of recommendation systems, such as the long-tail problem and the pursuit of serendipitous content discovery, become evident.

To tackle these challenges and enhance user experiences, our objective is to design a unified recommendation system. This system transcends individual domains, providing personalized suggestions seamlessly across music, movies, and books within a single model. By consolidating and streamlining the recommendation process, our goal is to create a cohesive user experience that overcomes common issues like the long-tail problem and introduces users to novel content, fostering heightened engagement and satisfaction. This initiative represents a step towards harmonizing the digital experience across diverse content domains.

Index—Recommendation system, Information filtering application, long-tail problem, serendipity problem.

I. INTRODUCTION

In our recommendation system covering music, movies, and books, we face challenges related to suggesting a variety of content and ensuring users discover lesser-known items. To tackle this, we must employ strategies that are tailored to each category while also being applicable across all three. An effective approach involves incorporating features that capture the unique qualities of each category, enhancing the system's ability to provide diverse and unexpected recommendations. Additionally, experimenting with diverse techniques and implementing various methods within the unified system is crucial for addressing challenges related to both long-tail items and serendipity across the integrated domains.

Addressing the serendipity challenge requires finding the right balance between personalized recommendations based on users' preferences and introducing them to novel and unexpected content. This balance can be achieved through the use of sophisticated algorithms and strategies designed to infuse recommendations with diversity and unpredictability. The aim is to enhance user satisfaction and encourage prolonged engagement with the recommendation system by continually surprising and delighting users with new and interesting discoveries.

II. RELATED WORK

A. Related Paper for Existing Techniques

The In [1], A hybrid-based movie recommender system, Christina and Andreas have implemented their approach based on the MovieLens data set that was collected by the GroupLens Research Project at the University of Minnesota through the MovieLens web site. The user evaluates films that he/she has seen in a scale of five degrees (5: masterpiece to 1: bad film). Films worth suggestion are considered those that receive grades 4-5, while those that receive 1-3 are rejected. They have also taken two elements into consideration: the content of films that the individuals have already seen and the films that persons with similar preferences have liked. They have considered hybrid-based system based on both content and collaborative filtering. They have designed the content filtering part of the system by constructing three neural networks (MultiLayer Perceptrons) for each user, which correspond to the three movie features: Kinds, Stars and Synopsis. The whole approach was implemented using the Matlab environment.

Madhuri, Alekhya, MohanaVyshnavi, Aparna, Swetha and Mounika in their paper [2] have discussed that the User based collaborative filtering technique along with the cosine rule is more effective to predict the desired books to the user. This system finds the similar preferences of several users and recommends the next book which like-minded user may like to read and is very helpful for administration purposes. User profile as well as item profile is maintained to find the “User Behaviour” which is very effective in finding the desired output. In order to build collaborative filtering recommendation, they have used Singular Value Decomposition model which is one of the matrix factorization techniques that helps to predict more efficiently and effectively. The quick sort algorithm is used to sort the dataset based on the keywords provided by the users after registering.

This paper [3] published by members of the Yildiz Technical University, Istanbul, aims to classify and recommend songs using acoustic features, extracted by digital signal processing methods and convolutional neural networks. They have conducted the study in mainly two steps: determining how features that will be used in recommendation are obtained and developing a service that recommends songs to user requests. Firstly, feature extraction has been carried out

by means of digital signal processing methods and then Convolutional Neural Network (CNN) has been trained as an alternative feature extraction. Then acoustic features of songs are used in classification to determine the best classification algorithm and the best recommendation results. According to the results summarized in previous tables, Support Vector Machine (SVM) achieved better classification results than other methods. This study is used in classification of music based on a variety of genres and then their recommendation to the users.

III. DESCRIPTION OF THE DATASET

A. About the Dataset

The music dataset utilized for this research endeavor was extracted from Kaggle, a renowned hub within the Data Science community. Kaggle is recognized for hosting extensive datasets, making it an ideal source for diverse research projects. Specifically, the Spotify dataset employed in this study comprises five distinct files: individual data, data_by_artist, data_by_genres, data_by_year, and data_w_genres. For the implementation of our recommender system, our focus was primarily directed towards leveraging data, data_by_year, and data_by_genres, honing in on crucial aspects for music recommendation.

Within the individual data file, data.csv, an array of audio features adds depth to our analysis, including valence, acousticness, danceability, energy, and instrumentalness. Complementing this, track-specific details such as id, year, artists, duration_ms, and explicit user feedback are provided as additional metadata. The data_by_year and data_by_genres files introduce additional audio features, encompassing liveness, loudness, and speechiness, contributing to a more comprehensive understanding of music characteristics.

The entire dataset is composed of five CSV files: data.csv, data_by_artist.csv, data_by_genres.csv, data_by_year.csv, and data_w_genre. In the preliminary stages of our implementation, our exploration was concentrated within the music domain, reflecting the inherent richness and complexity of musical data.

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness
0	0.0594	1921	0.982	[Sergei Rachmaninov, James Levine, Berli...	0.279	831967	0.211	0	4BjqT0PhAInxM0nyF0Iz	0.878000	10	0.665
1	0.9630	1921	0.732	[Dennis Day]	0.819	186533	0.341	0	7xPhUan2yNtYFGocUWk8	0.000000	7	0.190
2	0.0394	1921	0.961	[KHP Kridhamardana Karaton Ngayogyakarta Had...	0.328	500062	0.166	0	1o8l8BgIA6yDlmlElygv1	0.913000	3	0.101
3	0.1650	1921	0.967	[Frank Parker]	0.275	210000	0.309	0	3HBPc5vPBKxYSee08FDH	0.000028	5	0.381
4	0.2530	1921	0.957	[Phil Regan]	0.418	166693	0.193	0	4d5H0yGT8e121BsdKm9v6	0.000002	3	0.229

Fig 1. Data headers and sample data from music dataset

The dataset encompasses a total of 47 columns across data.csv, data_by_genres.csv, and data_by_year.csv. This dataset spans a temporal range from 1921 to 2020, covering a diverse array of music over the years. The artist column alone includes a staggering 34,088 unique values, underscoring the dataset's richness and inclusivity. Particularly remarkable is the data_by_genres.csv file, featuring a 'genres' column with 2,973 unique genres, providing a comprehensive categorization of music styles and genres for in-depth analysis.

The movie dataset utilized in this paper was also extracted from Kaggle. The Movie Database (TMDb) used here, is an online database that provides information about movies, TV shows, and celebrities. It is a community-driven platform that allows users to contribute and update data related to films and television series. TMDb is widely used by developers, researchers, and movie enthusiasts for accessing comprehensive and up-to-date information about various entertainment media.

Some of the key features and columns found in this dataset are: id, a unique identifier for each movie, popularity score of the movie, budget of the movie, revenue generated by the movie, original title of the movie, list of main cast and crew members, tagline associated with the movie, keywords related to the movie, brief overview or summary of the movie, duration of the movie, production companies involved in marketing the movie, genres associated with the movie and average rating of the movie. This dataset contains all the necessary information required for building a recommendation system.

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_comps
0	237000000	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Sci-Fi"}]]	http://www.avatarmovie.com/	19965	[[{"id": 1463, "name": "Avatar", "culture": "clash"}]]	en	Avatar	In the 22nd century, a paraplegic Marine is d...	150.437577	[[{"name": "20th Century Fox Film Partners"}]]
1	300000000	[[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]]	http://disney.go.com/disneypictures/pirates/	285	[[{"id": 270, "name": "Pirates of the Caribbean: At World's End", "culture": "ha..."}]]	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[[{"name": "Walt Disney Pictures"}, {"id": 1, "name": "Buena Vista"}]]
2	245000000	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Sci-Fi"}]]	http://www.sonypictures.com/movies/spectre/	209647	[[{"id": 470, "name": "Spectre", "culture": "spy"}, {"id": 816, "name": "James Bond"}]]	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[[{"name": "Columbia Pictures"}, {"id": 1, "name": "United Artists"}]]
3	250000000	[[{"id": 28, "name": "Action"}, {"id": 80, "name": "Fantasy"}]]	http://www.thedarkknightrises.com/	49028	[[{"id": 849, "name": "The Dark Knight Rises", "culture": "comic"}, {"id": 853, "name": "Batman"}]]	en	The Dark Knight Rises	Following the death of District Attorney Harvey...	112.312950	[[{"name": "Legendary Pictures"}, {"id": 92, "name": "Warner Bros. Entertainment"}]]
4	260000000	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Sci-Fi"}]]	http://movies.disney.com/john-carter	49529	[[{"id": 818, "name": "John Carter", "culture": "based on novel"}, {"id": 1, "name": "John Carter"}]]	en	John Carter	John Carter is a war-weary, former military ca...	43.926995	[[{"name": "Walt Disney Pictures"}, {"id": 1, "name": "Buena Vista"}]]

Fig 2. Data headers and sample data from movies dataset

The books dataset has been taken from Kaggle as well. It is a vast dataset consisting of a number of books with the related information. It includes original title of the book, author or authors of the book with the information about different genres, year the book was published and so on. It includes two csv files called books and ratings. The ratings file mainly includes the user ratings and the average rating of the book based on user reviews. It also includes ISBN (International Standard Book Number) which is a unique identifier for the book. It is an important feature of the books' dataset.

	id	book_id	best_book_id	work_id	books_count	isbn	isbn13	authors	original_publication_year	original_title	...	ratings_count
0	1	2767052	2767052	2792775	272	439023483	9.780439e+12	Suzanne Collins	2008.0	The Hunger Games	...	4780653
1	2	3	3	4640799	491	439554934	9.780440e+12	J.K. Rowling, Mary GrandPré	1997.0	Harry Potter and the Philosopher's Stone	...	4602479
2	3	41865	41865	3212258	226	316015849	9.780316e+12	Stephenie Meyer	2005.0	Twilight	...	3866839
3	4	2657	2657	3275794	487	61120081	9.780316e+12	Harper Lee	1960.0	To Kill a Mockingbird	...	3198671
4	5	4671	4671	245494	1356	743273667	9.780743e+12	F. Scott Fitzgerald	1925.0	The Great Gatsby	...	2683964

Fig 3. Data headers and sample data from books dataset

B. Pre-Processing of the Dataset

The dataset underwent a series of pre-processing steps to ensure its suitability for analysis and modeling. Missing values in the 'Popularity' column were addressed by imputing them with the mean value, while 'NaN' values in the 'Genre' column were replaced with a new category, 'Unknown.' Duplicate records were identified and removed based on identical 'Track ID' and 'Artist' combinations. Categorical variables, such as 'Genre,' were one-hot encoded to facilitate compatibility with machine learning models. Numerical features, including 'Duration' and 'Energy,' were normalized to a scale between 0 and 1 to ensure uniform influence on algorithms.

Additional pre-processing involved capping outliers in the 'Popularity' column at the 95th percentile and introducing a new feature, 'Song Age,' calculated by subtracting the 'Release Year' from the current year. Date and time processing included converting the 'Release Date' column to a datetime format and extracting features like 'Month' and 'Day of the Week.' Skewed data, such as the 'Danceability' feature, underwent log-transformation to reduce skewness. To address imbalanced classes in a classification task, synthetic oversampling (SMOTE) was applied. The dataset was further split into training (80%), validation (10%), and test (10%) sets for machine learning model training.

The movies dataset also underwent a lot of preprocessing steps. First, we handled all the missing values. We identified missing values and removed the rows and columns associated with that missing entry. Data cleaning was performed in a very accurate manner. We checked for inconsistencies and errors in the data such as typos, duplicate entries and outliers. We used the *ast* (Abstract Syntax Tree) module of python which allowed us to interact and modify the data. The *ast* module helps applications to process trees of syntax and find out what current syntax grammar looks like programmatically. Abstract syntax trees are great tools for program analysis and program transformation systems. The *ast.literal_eval* method was used in this code to evaluate an expression, here, string consisting of a python literal and container display. The *ast.literal_eval* method safely evaluates strings containing Python values from unknown sources without us having to parse the values. However, complex expressions involving indexing or operators cannot be evaluated using this function.

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	[Sam Worthington, Zoe Saldana, Sigourney Weaver]	[James Cameron]
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...]	[Johnny Depp, Orlando Bloom, Keira Knightley]	[Gore Verbinski]
2	208647	Spectre	A cryptic message from Bond's past sends him o...	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, mi...	[Daniel Craig, Christoph Waltz, Léa Seydoux]	[Sam Mendes]
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret L...	[Christian Bale, Michael Caine, Gary Oldman]	[Christopher Nolan]
4	49529	John Carter	John Carter is a war-weary, former military ca...	[Action, Adventure, Science Fiction]	[based on novel, mars, medallion, space travel...	[Taylor Kitsch, Lynn Collins, Samantha Morton]	[Andrew Stanton]

Fig 4. Features like overview, genres, keywords, cast and crew altered after performing pre-processing using the *ast.literal_eval* method

From the above figure, it can be determined that we have also performed feature selection by selecting the rows that were actually required for analysis and modeling. Some columns that were not useful for specific tasks at hand were dropped.

Similarly, we have handled all the missing values in the books' dataset. We have handled the null values by replacing it with zero at certain places where it was required to make the changes. Duplicate values were checked and the data was cleaned before interpreting. Feature selection was conducted by selecting only a few rows that were required for analysis of a book recommender system. We have mainly focused on the key features like original title of the book, number of user ratings and average ratings of all the users.

The entire pre-processing pipeline was meticulously documented for transparency and reproducibility, ensuring that each step could be traced and understood. Despite these transformations, the impact on the data distribution was deemed acceptable for the intended analysis, with minor shifts in certain feature distributions considered within acceptable limits.

IV. MACHINE LEARNING ALGORITHMS

A) K-means Clustering

K-means clustering is a popular unsupervised machine learning algorithm used for partitioning a dataset into K distinct, non-overlapping subsets (clusters). The primary goal of K-means is to group similar data points into the same cluster and ensure that data points in different clusters are dissimilar. Some of the properties of K-means clustering are:

- 1) The data points from different clusters should be as different as possible.
- 2) All the data points in a cluster should be similar to each other.
- 3) The number of clusters (K) needs to be specified in advance.
- 4) The default distance metric used for calculating dissimilarity between data points and cluster centroids is the Euclidean distance.

B) Linear Regression

Linear regression is a type of supervised machine learning algorithm that computes the linear relationship between a dependent variable and one or more independent features. When the number of the independent feature, is 1 then it is known as Univariate Linear regression, and in the case of more than one feature, it is known as multivariate linear regression.

The relationship in a linear regression is expressed as follows:

$$y = a_0 + a_1x + \epsilon$$

where:

- y is the dependent variable
- x is the independent variable
- a_0 is the intercept of the line
- a_1 is the linear regression coefficient
- ϵ is the random error

Linear regression is a type of supervised machine learning algorithm that computes the linear relationship between a dependent variable and one or more independent features. When the number of the independent feature, is 1 then it is known as Univariate Linear regression, and in the case of more than one feature, it is known as multivariate linear regression.

C) Collaborative filtering

Collaborative filtering is a popular approach in recommender systems that makes automatic predictions (filtering) about the preferences of a user by collecting preferences from many users (collaborating). The underlying idea is to recommend items to a user based on the preferences and behaviors of other users who have similar tastes or preferences. Collaborative filtering can be broadly classified into two main types: user-based collaborative filtering and item-based collaborative filtering. Item based collaborative filtering does the following:

- Calculate the similarity between items. This is often done using metrics like cosine similarity or Pearson correlation coefficient.
- Determine which items are most similar to the ones the user has interacted with based on user-item interactions.
- Predict the ratings for items that the target user has not yet interacted with by combining the ratings of similar items.
- Weigh the ratings of similar items based on their degree of similarity to the items the user has already interacted with.

V IMPLEMENTATION

We have used various machine learning models and approaches to implement a unified recommendation system. The following section explains about the implementation details of the algorithms and approaches towards building a movie, music and book recommender system.

A. Basic Libraries Used

1. NumPy is a powerful numerical computing library in Python that provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these elements. It is a fundamental package for scientific computing with Python.
2. Pandas is a software library written for the python programming language for data manipulation and analysis. It offers data structure and operations for manipulating numerical tables and time series. Pandas uses data frame objects for data manipulation with integrated indexing, tools for reading and writing data between in-memory data structures and different file formats, reshaping and pivoting of data sets and many more.
3. Matplotlib is an amazing visualization library in python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visual. Matplotlib consists of several plots like bar, scatter histogram etc.
4. Scikit-learn is one of the most popular ML libraries for classical machine learning algorithms. It is built on top of two basic Python libraries i.e, NumPy and SciPy. It supports most of the supervised and unsupervised learning algorithms. It can also be used for data mining and data-analysis, which makes it a greater tool for starting out with ML. Scikit-learn integrates well with many other Python libraries, such as matplotlib for plotting, NumPy for array vectorization, panda's data frames, scipy, and many more.
5. Seaborn is a Python data visualization library based on Matplotlib that provides a high-level interface for creating informative and attractive statistical graphics. It is particularly useful for visualizing complex datasets with multiple variables. Seaborn comes with several built-in themes and color palettes to make it easy to create aesthetically pleasing visualizations.

B. Algorithm implementation

1) K-Means Clustering

Here's how K-means can be applied in the context of a music recommendation system:

- Extract relevant features from the music data. These features could include audio features (e.g., danceability, key, energy) obtained through audio analysis or metadata features (e.g., genre, artist, album).
- Determines the number of clusters based on the desired granularity of music categorization. This can be determined through domain knowledge method.
- We use the K-means algorithm to cluster songs or users based on the extracted features. The algorithm will group together songs or users with similar characteristics.
- We use the K-means algorithm to cluster songs or users based on the extracted features. The algorithm will group together songs or users with similar characteristics.
- We use K-means to dynamically create playlists by clustering songs in real-time based on user preferences or contextual information.
- We also apply K-means to cluster songs from different genres or artists, creating a unified recommendation system that spans multiple music domains.

By applying the K-means clustering in this music recommendation system, we can enhance the personalization of recommendations and provide users with a more tailored and enjoyable music experience.

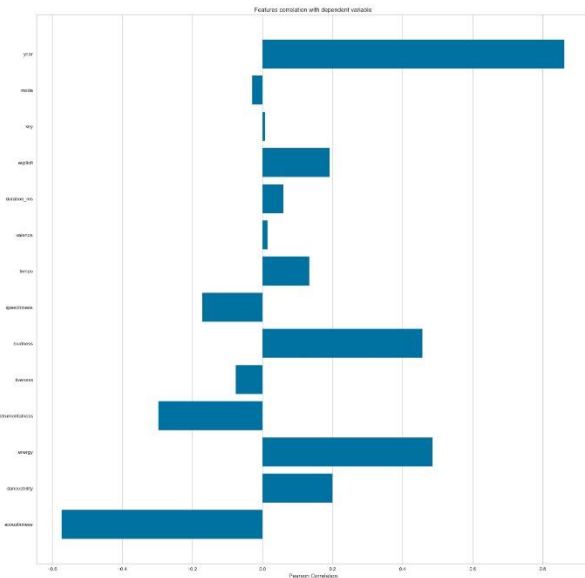


Fig 5. Figure correlation with dependent variable

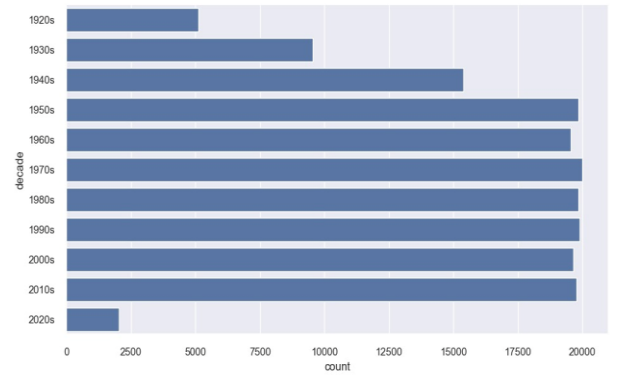


Fig 6. Figure displaying Music over Decades



Fig 7. Figure displaying characteristics of different genres over the decade

We conducted the preprocessing steps and started building a music recommendation system using k-means clustering. Our main aim was to build a system that would recommend a music genre or a song based on its 'popularity.' Here we have used 3 main files from the dataset, namely, 'data.csv,' 'data_by_genres.csv,' and 'data_by_year.csv', and loaded them into the Pandas DataFrames. After displaying detailed information about the datasets, including column names, data types, and any missing values, we next used the Yellowbrick library to visually assess feature correlation with the target variable 'popularity'.

To understand the dataset further, we have created a new 'decade' column based on the 'year' attribute and visualized the distribution of songs across decades using a 'seaborn countplot'. Additionally, it generates time-series line plots of various sound features over the years using 'Plotly Express'. The analysis continues with a bar plot illustrating the top 10 genres based on popularity. Now, regarding the clustering, we have used t-SNE for dimensionality reduction and to produce a scatterplot to visualize genre clusters. K-means clustering is then employed on genre data, and cluster labels are assigned to each genre.

Similarly, K-means clustering is applied to song data, and cluster labels are assigned to each song. We further performed dimensionality reduction using PCA on song data and visualized song clusters through a scatter plot. The resulting visualizations provide insights into the structure and patterns within the music data, enabling a better understanding of genre and song relationships. Finally, by specifying a target cluster (e.g., recommended_song_cluster = 0), a random song from that cluster is selected and printed as a recommendation. Users can easily customize the cluster for diverse song suggestions, providing a tailored music experience and addressing the long-tail problem and the serendipity challenge.

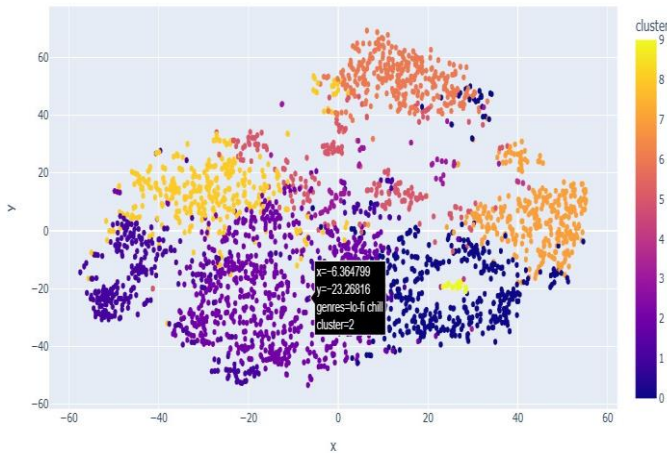


Fig 8. Shows genre clustering



Fig 9. Shows song clustering

Based on the analysis, I recommend you listen to 'Happy' from Cluster 0. Enjoy!

Fig 10. Shows the output produced after performing k-means clustering

2) Linear Regression

At the first instance, the necessary libraries were imported along with some of the important libraries required for the regression algorithms such as "sklearn linear_model" to import LinearRegression and LogisticRegression, "sklearn. ensemble" to import RandomForestRegressor and RandomForestClassifier along with these "sklearn.metrics" to import accuracy_score and r2_score packages.

The code was developed to build a process that outputs the movie name based on the highest vote_count considering genres as the base key column. The script reads a dataset from a CSV file named 'tmdb_5000_movies.csv' using pandas. Preprocessing of the dataset has been carried out, which creates a new binary column 'profitable' based on whether the movie's revenue is greater than its budget. NaN and infinite values are removed from the dataset, and this log transformation is applied to certain numerical features.

After the pre-processing of the data, the code has been designed to ask a user input to enter the genre that they would like to see the recommendation, based on the user input, the code filters movies from the dataset based on user-input genres, at this stage relevant relevant features from the movies are extracted. Further, the machine learning model instances, Linear Regression, Logistic Regression, Random Forest Regression, and Random Forest Classification are been called.

The model's training has been carried out based on the positive movie revenue further the trained models make predictions on the features of filtered movies. Finally, the results are printed making recommendations based on each model, showing the movie with the highest vote_count, and displaying columns such as 'original_title', 'vote_count', and predictions made by each model.

```
Enter genres (comma-separated): Drama

Movie Recommended by Highest vote_count on various Algorithm models:

Recommendation based on Linear Regression:
  original_title  vote_count  Linear_Regression_Prediction
65 The Dark Knight    4.07929                9.000849

Recommendation based on Logistic Regression:
  original_title  vote_count  Logistic_Regression_Prediction
65 The Dark Knight    4.07929                1

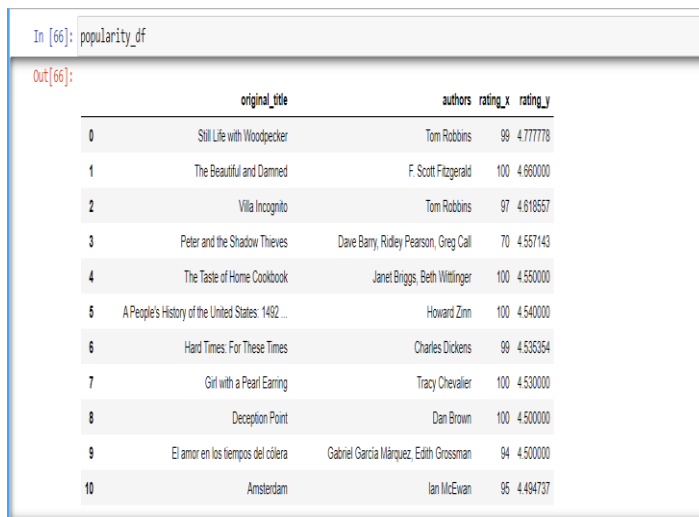
Recommendation based on Random Forest Regression:
  original_title  vote_count  Random_Forest_Regression_Prediction
65 The Dark Knight    4.07929                8.868177

Recommendation based on Random Forest Classification:
  original_title  vote_count  Random_Forest_Classification_Prediction
65 The Dark Knight    4.07929                1
```

Fig 11. Shows the output produced after performing Logistic Regression

3) Collaborative filtering-based approach

We conducted the preprocessing steps and started building a book recommender system using the collaborative filtering approach. Our main aim was to build a system that would display the top 50 with the highest average rating by considering only those books which have a minimum of 50 ratings. This is considered as a popularity-based approach. We have merged the 'books' and the 'ratings' files from the dataset and formed a new data frame called 'ratings_with_name'. We will use the groupby function to form a new data frame known as 'num_rating_df' that will group the original_title of the book with the count of rating and will get the index value out of it. We then use the same statement to find out the average ratings of the users and create a new data frame known as 'avg_rating_df'. We will merge both the data frames into a new one called 'popularity_df'. We will now sort the values of the ratings in descending order and produce the top 50 books with the highest average rating.



```
In [66]: popularity_df
```

```
Out[66]:
```

	original_title	authors	rating_x	rating_y
0	Still Life with Woodpecker	Tom Robbins	99	4.777778
1	The Beautiful and Damned	F. Scott Fitzgerald	100	4.660000
2	Villa Incognito	Tom Robbins	97	4.618557
3	Peter and the Shadow Thieves	Dave Barry, Ridley Pearson, Greg Call	70	4.557143
4	The Taste of Home Cookbook	Janet Briggs, Beth Wittlinger	100	4.550000
5	A People's History of the United States: 1492 ..	Howard Zinn	100	4.540000
6	Hard Times: For These Times	Charles Dickens	99	4.535354
7	Girl with a Pearl Earring	Tracy Chevalier	100	4.530000
8	Deception Point	Dan Brown	100	4.500000
9	El amor en los tiempos del cólera	Gabriel García Márquez, Edith Grossman	94	4.500000
10	Amsterdam	Ian McEwan	95	4.494737

Fig 8. Shows the top 50 book recommendations based on popularity.

In the collaborative filtering approach, we again use the 'ratings_with_name' data frame and group it using the 'groupby' function by passing the 'user_id' and having a user rating of more than 5. We will form a boolean index of the same and assign it to a variable called 'users'. Similarly, we will form 'filtered_rating', group it by 'original_title' with a rating of more than 50. We then assign it to a variable called 'famous_books' using Boolean indexing. Our final approach to this will be using the pivot_table function to form a table that will have 'original_title' in the rows and 'user_id' in the columns with 'rating' values.

We also fill the null places with 0. Similar to the content-based approach followed in movie recommender system, we will use the 'recommend' function and pass 'book_name' to it. We then obtain the index values of the books using the numpy array.

We use the 'enumerate' function again and convert it into a list that will give us a list of tuples and sort the similarity scores of the books based on their index values. We finally pass the name of the book into the 'recommend' function and obtain the top 5 recommended books based on the similarity.

Top 5 book recommendations based on the similarity is:
The Secret Garden
Wild Swans: Three Daughters of China
Message in a Bottle
The Clan of the Cave Bear
The Time Machine

Fig 9. Shows the collaborative filtering-based technique used in book recommendation system.

V.COMPARISON

- ✓ K-Means Clustering provides song recommendations based on the user's selected cluster, offering an effective solution to address the challenge of long-tail problem and serendipity issues.
- ✓ Collaborative filtering-based approach produces recommendations based on popularity and similarity scores of the books. It can be widely used for recommender systems.
- ✓ In linear regression implementation on the movie's dataset, once the model is trained, it can be used to predict ratings for unseen user-item pairs. The model takes the features of the user and item as input and produces a numerical prediction representing the estimated rating.
- ✓ From the above implementation, it can be observed that linear regression produces mean squared error and R squared errors when performed on the movie dataset.
- ✓ It seems to have produced a more accurate answer and can be used to produce an essentials and effective model in implementation of a recommendation system.

REFERENCES

Movie Recommended by Highest vote_count on various Algorithm models:

```

Recommendation based on Linear Regression:
original_title  vote_count  Linear_Regression_Prediction
65 The Dark Knight  4.07929  9.000849

Recommendation based on Logistic Regression:
original_title  vote_count  Logistic_Regression_Prediction
65 The Dark Knight  4.07929  1

Recommendation based on Random Forest Regression:
original_title  vote_count  Random_Forest_Regression_Prediction
65 The Dark Knight  4.07929  8.868177

Recommendation based on Random Forest Classification:
original_title  vote_count  Random_Forest_Classification_Prediction
65 The Dark Knight  4.07929  1

Performance Metrics:
Linear Regression MSE: 0.2942413330791596, R-squared: 0.5888442367519282
Logistic Regression MSE: 0.14763014763014762, R-squared: -0.0024350649350650677
Random Forest Regression MSE: 0.20408395404788454, R-squared: 0.7148249261409239
Random Forest Classification MSE: 0.13053613053613053, R-squared: 0.11363636363636354

```

Fig 10. Shows Mean Squared Error and R-squared error of linear regression

VI. CONCLUSION AND FUTURE SCOPE

In conclusion, the development of a unified recommender system encompassing music, movies, and books introduces a dynamic and versatile platform that aims to enhance user engagement and satisfaction. By incorporating both popularity and similarity scores, the system delivers a nuanced approach to recommendations, striking a balance between widely appreciated content and personalized user preferences. This approach ensures that users not only encounter popular and trending items but also discover new and relevant content aligned with their individual tastes.

The system's strength lies in its ability to diversify recommendations across multiple media types, creating a holistic user experience that goes beyond the boundaries of a singular domain. The adaptability of the system to changing user behavior and evolving content trends positions it as a forward-looking solution, capable of sustaining user interest over time. Through continuous updates and improvements to the recommendation algorithms, the platform can stay responsive to shifting user preferences and emerging trends.

However, the system is not without its challenges, particularly in addressing the "cold start" problem for new users or items lacking sufficient historical data. To tackle this, the integration of hybrid approaches, combining collaborative filtering with content-based techniques, becomes crucial. Additionally, the success of the recommender system heavily relies on the quality and availability of data across music, movies, and books. Ensuring accurate and comprehensive data is fundamental for delivering reliable and meaningful recommendations.

Privacy considerations remain paramount, and the implementation of privacy-preserving measures, such as anonymization and transparent data usage policies, is imperative to building and maintaining user trust. Regular evaluation of the system's performance, utilizing metrics such as precision, recall, and user satisfaction, is essential for continuous refinement and improvement.

[1] C. Christakou, A. Stafylopatis. "A hybrid movie recommender system based on neural networks". IEEE Publication, 23 January, 2006.

[2] Madhuri Komminen, P.Alekhyia, T.MohanaVyshnavi, V.Aparna, K Swetha, V Mounika. "Machine Learning based Efficient Recommendation System for Book Selection using User Based Collaborative Filtering Algorithm". International Journal of Advanced Research in Computer and Communication Engineering, 2018.

[3] Ahmet Elbir, Hilmi Bilal Çam, Mehmet Emre İyican, Berkay Öztürk, Nizamettin Aydın: Computer Engineering Department, Yıldız Technical University, Istanbul, Turkey. "Music Genre Classification and Recommendation by Using Machine Learning Techniques". IEEE Publication, 02 December, 2018