MCA Semester – IV

Research Project – Interim Report

Name	ARSHA P JOY
Project	Movie Recommendation System
Group	Individual
Date of Submission	16/04/2024

A study on "Movie Recommendation System"

Research Project submitted to Jain Online (Deemed-to-be University)

In partial fulfillment of the requirements for the award of:

Master of Computer Application

Submitted by:

ARSHA P JOY

USN:

221VMTR01438

Under the guidance of:

S Gayathri - JAIN Online

Jain Online (Deemed-to-be University)

Bangalore

2023-24

DECLARATION

I, *Arsha P Joy*, hereby declare that the Research Project Report titled "**Movie Recommendation System***"* has been prepared by me under the guidance of *S Gayathri.* I declare that this Project work is towards the partial fulfillment of the University Regulations for the award of the degree of Master of Computer Application by Jain University, Bengaluru. I have undergone a project for a period of Eight Weeks. I further declare that this Project is based on the original study undertaken by me and has not been submitted for the award of any degree/diploma from any other University / Institution.

Place: Bangalore

Date: 16/04/2024 *Arsha P Joy*

USN: 221VMTR01438

List of Figures		
No.	**Title**	**Page No.**
1	Dataset View	9
2	Data Info	9
3	Missing values	10
4	Percentage of missing values	10
5	Descriptive statistics	12
6	Clean data view	12
7-a	Distribution of audience scores	13
7-b	Distribution of tomato meter Scores	13
8-a	Distribution of movie runtimes	13
8-b	Distribution of movie ratings	13
9	Distribution of review score sentiment	14
10-a	Top 10 most frequent genres	14
10-b	Top 20 most frequent languages	14
11-a	Top 10 directors with most movies	15
11-b	Top 10 writers with most movies	15
12-a	Top 10 distributors with most movies	15
12-b	Most frequent words in review texts	15
13	Recommendations	18

Table of Contents	
Chapter	**Topics**
Objectives of the Study	Defining problem statement
	Need of the study/project
Scope of the Study	Understanding business/social opportunity
Data Collection Method	Data Overview and details
Data Analysis Tools	1. Exploratory data analysis
	2. Business insights from EDA
Methodology	Methodology of the recommendation system

Interim Report: Movie Recommendation System Project

Objectives of the Study

Defining the Problem Statement:

Movie recommender systems have become a hot topic in recent years due to their significant impact on the entertainment industry and user experiences. With the vast number of movies available across various platforms, it can be challenging for viewers to navigate and discover films that align with their tastes. Movie recommender systems leverage advanced algorithms and data analysis techniques to offer personalized recommendations based on user preferences, viewing history, ratings, and other relevant factors.

These recommendations not only help viewers save time and effort in searching for movies but also enhance their overall entertainment experience. By providing tailored suggestions, movie recommender systems expose users to a broader range of films, including hidden gems and lesser-known titles, which they might not have otherwise come across. This not only encourages diversity in movie consumption but also supports filmmakers and content creators by promoting a wider viewership.

The objective here is to build a content-based movie recommender system. The system should analyze any/all movie features, such as audience scores, genres, directors, and critic reviews, to provide movie recommendations.

Need of the Study/Project:

In today's vast entertainment landscape, users often face difficulty in discovering movies that align with their tastes and preferences. Existing recommendation systems may not adequately cater to individual preferences or promote diversity in movie consumption. Hence, there is a need for a

content-based recommendation system that leverages movie features to offer personalized recommendations.

Scope of the Study

Understanding Business/Social Opportunity:

The project seeks to address the challenge of movie discovery by developing a content-based recommendation system. By analyzing various movie features such as audience scores, critic reviews, genres, directors, and ratings, the system aims to offer tailored recommendations to users. This will not only enhance user experience but also promote diversity in movie consumption by exposing users to a broader range of films.

Data Collection Method

The data was taken from Rotten tomatoes an American review-aggregation website for film and television. There were two datasets available, and the features are listed below for each dataset:

1. Rotten Tomatoes Movies Dataset: rotten_tomatoes_movies.csv

- id: Unique identifier for each movie.
- title*:* The title of the movie.
- audienceScore: The average score given by regular viewers.
- tomatoMeter: The percentage of positive reviews from professional critics.
- rating: The movie's age-based classification (e.g., G, PG, PG-13, R).
- ratingContents: Content leading to the rating classification.
- releaseDateTheatres: The date the movie was released in theaters.
- releaseDateStreaming: The date the movie became available for streaming.

- runtimeMinutes: The duration of the movie in minutes.
- genre: The movie's genre(s).
- originalLanguage: The original language of the movie.
- director: The movie's director.
- writer: The writer(s) responsible for the movie's screenplay.
- boxOffice: The movie's total box office revenue.
- distributor: The company responsible for distributing the movie.
- soundMix: The audio format(s) used in the movie.

2. Rotten Tomatoes Movie Reviews Dataset: rotten_tomatoes_movie_reviews.csv

- id: Unique identifier for each movie (matches the id in rotten_tomatoes_movies.csv).
- reviewId: Unique identifier for each critic review.
- creationDate: The date the review was published.
- criticName: The name of the critic who wrote the review.
- isTopCritic: A boolean value indicating if the critic is considered a top critic.
- originalScore: The score provided by the critic.
- reviewState: The status of the review (e.g., fresh, rotten).
- publicatioName: The name of the publication where the review was published.
- reviewText: The full text of the critic review.
- scoreSentiment: The sentiment of the critic's score (e.g., positive, negative, neutral).
- reviewUrl: The URL of the original review on Rotten Tomatoes.

Figure 1

	id	title	audienceScore	tomatoMeter	rating	ratingContents	releaseDateTheaters	releaseDateStreaming	runtimeMinutes	genre	...	reviewId	creationDate	criticName	isTopCritic	or:
0	space-zombie-bingo	Space Zombie Bingo!	50.0	NaN	NaN	NaN	NaN	2018-08-25	75.0	Comedy, Horror, Sci-fi	...	NaN	NaN	NaN	NaN	
1	the_green_grass	The Green Grass	NaN	NaN	NaN	NaN	NaN	2020-02-11	114.0	Drama	...	NaN	NaN	NaN	NaN	
2	love_lies	Love, Lies	43.0	NaN	NaN	NaN	NaN	NaN	120.0	Drama	...	2739073.0	2020-10-31	James Mudge	False	
3	love_lies	Love, Lies	43.0	NaN	NaN	NaN	NaN	NaN	120.0	Drama	...	2333658.0	2016-06-15	Diva Velez	False	
4	the_sore_losers_1997	Sore Losers	60.0	NaN	NaN	NaN	NaN	2020-10-23	90.0	Action, Mystery & thriller	...	NaN	NaN	NaN	NaN	

5 rows x 26 columns

The combined Data from both the Dataset files are shown in Figure 1. There are 1543226 rows and 26 columns. The dataset contains several columns providing information such as title, audienceScore, tomatoMeter, rating, genre, director, etc. There are missing values (NaN) in some columns, which need to be handled later during preprocessing. The genre column seems to contain multiple genres separated by commas, which required further processing to use genre information for recommendation. There are some other columns which also needs text processing.

Data Analysis Tools

Exploratory Data Analysis

Figure 2

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1543226 entries, 0 to 1543225
Data columns (total 26 columns):
 #   Column                Non-Null Count    Dtype
---  ------                --------------    -----
 0   id                    1543226 non-null  object
 1   title                 1537277 non-null  object
 2   audienceScore         1421875 non-null  float64
 3   tomatoMeter           1398498 non-null  float64
 4   rating                892146 non-null   object
 5   ratingContents        892146 non-null   object
 6   releaseDateTheaters   1189667 non-null  object
 7   releaseDateStreaming  1387046 non-null  object
 8   runtimeMinutes        1507064 non-null  float64
 9   genre                 1514640 non-null  object
 10  originalLanguage      1506604 non-null  object
 11  director              1530681 non-null  object
 12  writer                1354349 non-null  object
 13  boxOffice             1010898 non-null  object
 14  distributor           1136182 non-null  object
 15  soundMix              706721 non-null   object
 16  reviewId              1469840 non-null  float64
 17  creationDate          1469840 non-null  object
 18  criticName            1469840 non-null  object
 19  isTopCritic           1469840 non-null  object
 20  originalScore         1026937 non-null  object
 21  reviewState           1469840 non-null  object
 22  publicatioName        1469840 non-null  object
 23  reviewText            1399555 non-null  object
 24  scoreSentiment        1469840 non-null  object
 25  reviewUrl             1255323 non-null  object
dtypes: float64(4), object(22)
memory usage: 306.1+ MB
```

Data Frame has a shape of (1543226, 26), indicating that it contains a total of 1543226 rows and 26 columns. This suggests that the merger operation has resulted in a larger dataset, The information from Figure 2 shows that there are a few wrong datatypes and figure 3 and 4 shows the missing values and the percentage of missing values.

Figure 3

```
id                         0
title                   5949
audienceScore         121351
tomatoMeter           144728
rating                651080
ratingContents        651080
releaseDateTheaters   353559
releaseDateStreaming  156180
runtimeMinutes         36162
genre                  28586
originalLanguage       36622
director               12545
writer                188877
boxOffice             532328
distributor           407044
soundMix              836505
reviewId               73386
creationDate           73386
criticName             73386
isTopCritic            73386
originalScore         516289
reviewState            73386
publicatioName         73386
reviewText            143671
scoreSentiment         73386
reviewUrl             287903
dtype: int64
```

Figure 4

```
Percentage of missing values:
 id                     0.000000
title                  0.385491
audienceScore          7.863463
tomatoMeter            9.378276
rating                42.189543
ratingContents        42.189543
releaseDateTheaters   22.910384
releaseDateStreaming  10.120358
runtimeMinutes         2.343273
genre                  1.852353
originalLanguage       2.373081
director               0.812908
writer                12.239102
boxOffice             34.494494
distributor           26.376176
soundMix              54.204958
reviewId               4.755363
creationDate           4.755363
criticName             4.755363
isTopCritic            4.755363
originalScore         33.455178
reviewState            4.755363
publicatioName         4.755363
reviewText             9.309784
scoreSentiment         4.755363
reviewUrl             18.655919
dtype: float64
```

This information is valuable for determining how to handle missing values during preprocessing. For columns with a high percentage of missing values, it's essential to carefully consider their relevance to the recommendation system and whether imputation or exclusion is appropriate.

There are so many missing values to be treated using various methods including dropping the unnecessary features and features with more than 50 % missing values, filling, and creating new category etc.

Handling the Duplicates and missing values

There were 2.44% of duplicate rows and 5949 rows with missing titles which were removed using the drop() method.

'releaseDateTheaters', 'releaseDateStreaming', 'soundMix', 'reviewId', 'creationDate', 'criticName', 'isTopCritic', 'reviewState', 'publicatioName', 'reviewUrl' were the unnecessary columns which also removed from further processing.

The missing values at 'audienceScore', 'tomatoMeter', 'runtimeMinutes' were replaced with mean imputation.

For the 'boxOffice', there are values with '$' and "K", "M" as suffixes and its datatype is object but the values are numeric. So, the symbols were removed and converted the values to million dollars and then the datatypes is changed into float without scientific notation and assigned to a new variable named 'boxOffice_numerical'.

'originalScore' had object datatype and had different scales of rating. So, a custom function is used to normalize original scores to a scale of 10 and assigned to a new variable named 'originalScore_normalized'. Then the missing values at both "boxOffice_numerical" and 'originalScore_normalized' are filled with mean imputation method.

'rating', 'ratingContents', 'genre', 'originalLanguage', 'director', 'writer', 'distributor', 'reviewText', 'scoreSentiment' are important features and the missing values were handled by filling a new category 'unknown' since all of them are object datatype and filling with a mean/median/mode is not a good approach. Figure 5 shows the descriptive statistics of the numerical features.

Figure 5

	audienceScore	tomatoMeter	runtimeMinutes	boxOffice_numerical	originalScore_normalized
count	1.499702e+06	1.499702e+06	1.499702e+06	1.499702e+06	1.499702e+06
mean	6.497196e+01	6.692034e+01	1.063748e+02	4.796943e+01	4.634202e+01
std	1.913096e+01	2.484738e+01	2.180480e+01	7.155819e+01	2.721925e+04
min	0.000000e+00	0.000000e+00	1.000000e+00	0.000000e+00	-2.500000e+00
25%	5.200000e+01	5.100000e+01	9.300000e+01	4.100000e+00	6.000000e+00
50%	6.600000e+01	7.000000e+01	1.040000e+02	4.796943e+01	8.750000e+00
75%	8.000000e+01	8.800000e+01	1.170000e+02	4.796943e+01	4.634202e+01
max	1.000000e+02	1.000000e+02	2.700000e+03	8.584000e+02	3.333334e+07

'scoreSentiment' has categories such as 'Unknown', 'POSITIVE', 'NEGATIVE' and 'rating' has categories like 'nan', 'PG-13', 'TVPG', 'R', 'PG', 'TV14', 'NC-17', 'TVG', 'TVMA', 'TVY7', 'G' which need to be handled by encoding. Also, there are text values with various symbols and stop words which also need to be handled while processing. Figure 6 shows the data after this much cleaning.

Figure 6

	id	title	audienceScore	tomatoMeter	rating	ratingContents	runtimeMinutes	genre	originalLanguage	director	writer	distributor	reviewText	scoreSentiment	boxOffice_numeri
0	space-zombie-bingo	Space Zombie Bingo!	50.000000	66.920339	Unknown	Unknown	75.0	Comedy, Horror, Sci-fi	English	George Ormrod	George Ormrod,John Sabotta	Unknown	Unknown	Unknown	47.969
1	the_green_grass	The Green Grass	64.971958	66.920339	Unknown	Unknown	114.0	Drama	English	Tiffany Edwards	Tiffany Edwards	Unknown	Unknown	Unknown	47.969
2	love_lies	Love, Lies	43.000000	66.920339	Unknown	Unknown	120.0	Drama	Korean	Park Heung-Sik,Heung-Sik Park	Ha Young-Joon,Jeon Yun-su,Song Hye-jin	Unknown	Though let down by its routine love triangle n...	POSITIVE	47.969
3	love_lies	Love, Lies	43.000000	66.920339	Unknown	Unknown	120.0	Drama	Korean	Park Heung-Sik,Heung-Sik Park	Ha Young-Joon,Jeon Yun-su,Song Hye-jin	Unknown	While not perfect, Love, Lies is a worthy disc...	POSITIVE	47.969
4	the_sore_losers_1997	Sore Losers	60.000000	66.920339	Unknown	Unknown	90.0	Action, Mystery & thriller	English	John Michael McCarthy	John Michael McCarthy	Unknown	Unknown	Unknown	47.969

Figure 7-a, Figure 7-b

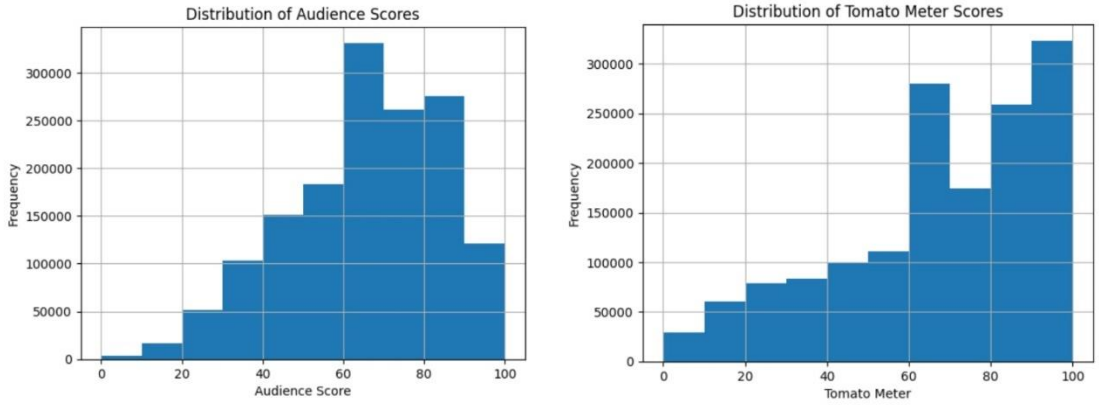


Figure 8-a, Figure 8-b

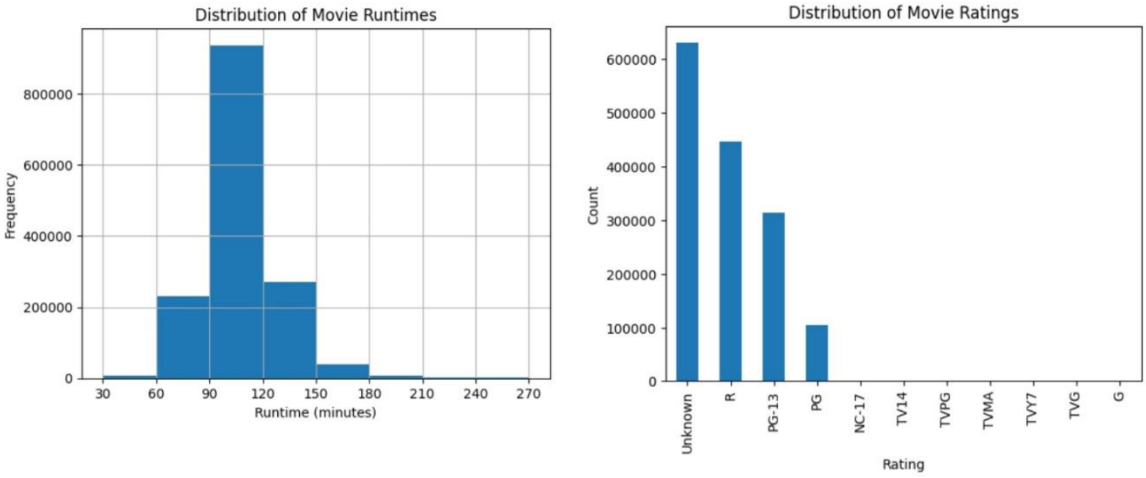


Figure 7-a shows the distribution of audience score and 7-b shows the tomatoMeter scores. The most scores are under the range of 40-100. Figure 8-a shows the distribution of movie runtime and 8-b shows the distribution of movie ratings. Most of the movies have a runtime range between 90-120 and the R category is the most rated category followed by PG-13 and PG. Figure 9 shows the distribution of review score sentiment and the positive reviews are higher than the negatives.

Figure 9

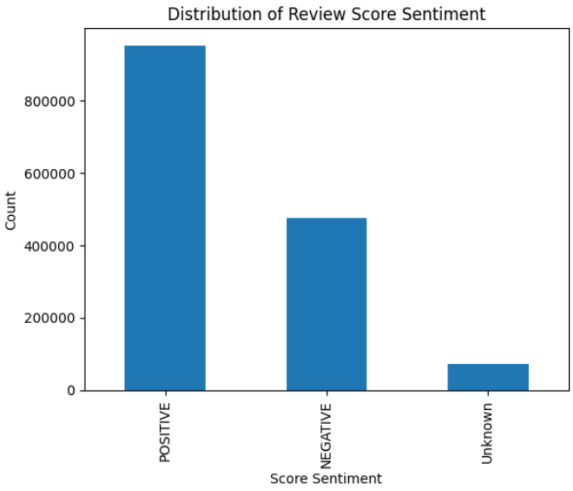
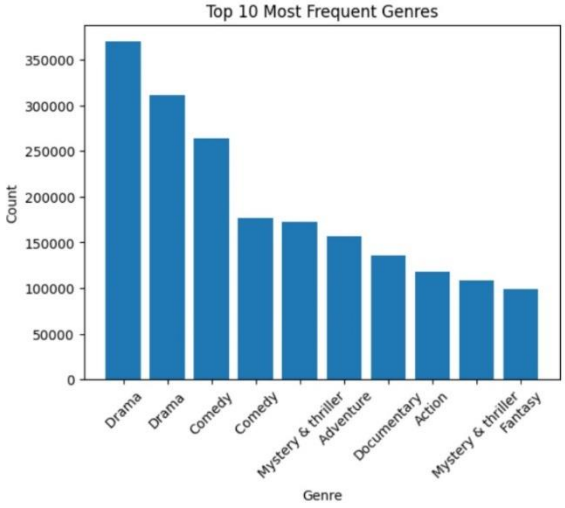


Figure 10-a, Figure 10-b



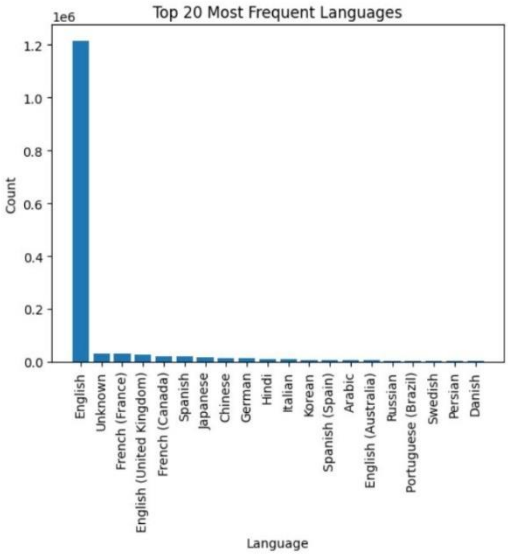
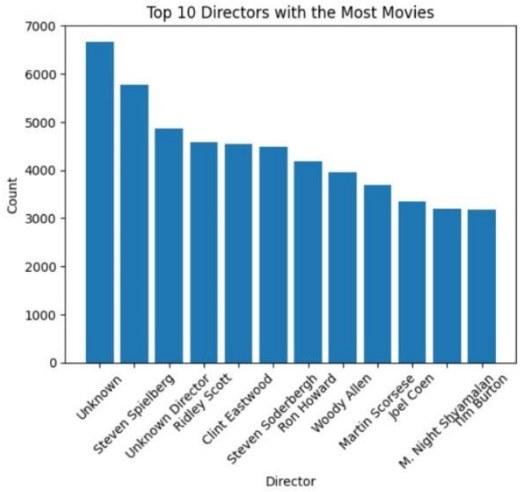


Figure 11-a, Figure 11-b



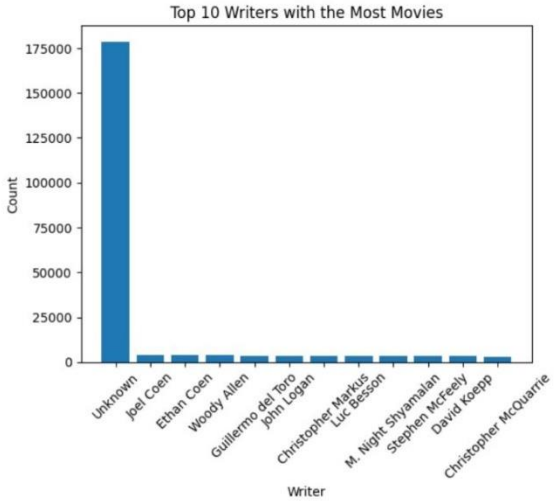
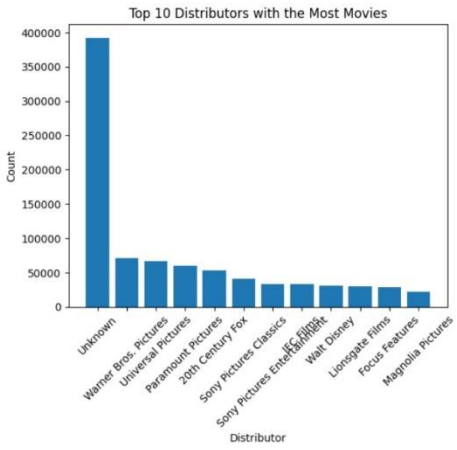


Figure 12-a, Figure 12-b



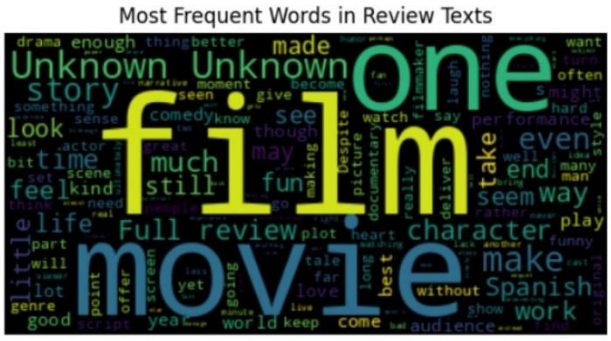


Figure 10-a,b,11-a,b,12-a,b shows the most frequent genres, frequent languages, Directors, writers, and distributers with most movies and the most frequent words in reviews.

Business insights from EDA

- Drama, Comedy, Mystery & Thriller, Adventure, Documentary, Action, Fantasy are the most frequent genres of the movies.
- English, French, English (UK), French (Canada), Spanish, Japanese, Chinese German, Hindi, and Italian are the top frequent languages of the movies.
- Steven Spielberg, Ridley Scott, Clint Eastwood, Steven Soderbergh, Ron Howard are the Directors with the greatest number of movies.
- Joel Coen, Ethan Coen, Woody Allen, Guillermo del Toro, John Logan are the writers with most movies.
- Warner Bros. Pictures, Universal Pictures, 20th Century Fox, Sony Pictures Classics, IFC Films are the distributors with a greater number of movies.
- Film, movie, one, made, story, character, look, make, take, way, made, fun were some common words frequently appeared in the reviews.

Methodology

'ratingContents', 'genre', 'director', 'writer', 'distributor', 'reviewText' were the columns with text data and haven't yet processed. It contains many stop words and symbols. 'rating', 'scoreSentiment' have categorical values and 'audienceScore', 'tomatoMeter', 'runtimeMinutes', 'boxOffice_numerical', 'originalScore_normalized' are numerical columns. Three lists are defined to categorize the columns in the dataset: text_columns for textual data, categorical_columns for categorical data, and numerical_columns for numerical data.

The TfidfVectorizer and LabelEncoder objects are initialized to transform text and categorical data, respectively. The 'title' column in the dataset is encoded using LabelEncoder to convert movie titles into numerical representations. To handle memory constraints and improve performance, text columns are processed in batches. For each batch, text data is transformed using TfidfVectorizer, filling NaN values with empty strings, and converting them to string data type if necessary.

The resulting sparse matrices are converted to dense arrays and updated in the original dataframe. Similarly, categorical columns are processed in batches using LabelEncoder to encode categorical labels into numerical representations. Numerical columns are standardized using StandardScaler, which scales each feature to have a mean of 0 and a standard deviation of 1. A content-based movie recommendation system is built using the TF-IDF (Term Frequency-Inverse Document Frequency) algorithm and cosine similarity.

Libraries from scikit-learn for text vectorization (TfidfVectorizer) and computing pairwise similarity scores (linear kernel) is used. To reduce computational complexity, a smaller subset of the dataset is loaded, containing the first 10,000 rows. This smaller subset helps in faster prototyping and testing.

Duplicate reviews based on the combination of movie title and review text are removed to ensure uniqueness in the dataset. TfidfVectorizer is defined with stop words removed, which will convert textual data into TF-IDF vectors. The 'reviewText' column is fit and transformed using TfidfVectorizer to obtain TF-IDF matrices representing each review. Using the linear_kernel function, the cosine similarity matrix is computed from the TF-IDF matrix. This matrix represents the similarity between each pair of reviews based on their TF-IDF vectors. The content_based_recommendations function takes a movie title as input and returns a list of recommended movies based on their similarity to the input movie. The input movie title is processed by removing leading and trailing whitespace and converted to lowercase for case-insensitive matching. The function retrieves the index of the input movie in the dataset and then obtains the similarity scores of all movies with respect to the input movie based on the cosine similarity matrix. The function sorts the movies based on similarity scores and filters out duplicate and input movie titles from the recommendations. Finally, an example usage of the content based recommendation function is demonstrated by providing the movie title "Adrift" and printing the recommended movies.

Figure 13

```
1                                        The Green Grass
2                                             Love, Lies
4                                            Sore Losers
5                                        Dinosaur Island
46                                        Scrambled Beer
47    Kakabakaba ka ba? (Will Your Heart Beat Faster?)
48                                             Sundowning
49                                          Born to Kill
55                             Number One With a Bullet
56                               The Garden Murder Case
Name: title, dtype: object
```

10 recommendations are generated for the movie Adrift, and the recommendations need to be optimized and tuned for better results.