ID3801 - Open Ended Lab Project
Development of a recommender system on IMDB Movie dataset

Sasidhar Reddy Navuluri
Venkata Karthikeya Mantripragda

Abstract

This project focuses on the development and implementation of a movie recommendation system content-based filtering techniques and matrix factorization. The dataset comprises diverse movie features, including genres, keywords, cast popularity, director popularity, writer popularity, and story elements. The preprocessing steps involve data cleaning, text analysis, and feature normalization. The matrices representing each movie's characteristics are then subjected to Non-negative Matrix Factorization (NMF) for dimensionality reduction. The resulting latent features are utilized to calculate cosine similarity, enabling the system to recommend movies based on their similarity to a user's preferences. The report details the data processing pipeline, matrix factorization, and the recommendation generation process. The system is evaluated using a real-world example, showcasing its effectiveness in providing relevant movie recommendations.

1. Problem Statement

Develop a movie recommendation system based on a dataset containing information about movies, including genres, cast, directors, writers, keywords, and story details. The goal is to create a content-based recommendation system and Matrix factorization that suggests movies similar to a given movie.

2. Methodology

2.1. Overview

1. Data Collection
2. Data Preprocessing
   - Handling Missing Values
   - Dealing With Categorical Data
   - Preprocessing Storyline Data
3. Combining Features
4. Matrix Normalization

2.2. Data collection

The script uses web scraping techniques with BeautifulSoup and Selenium to extract movie information from IMDb. It iterates through a list of movie titles, retrieves IMDb URLs using Selenium, and then scrapes relevant details such as keywords, genres, directors, writers, cast,story and the popularity rankings of the directors,cast,writers. The data is stored in a CSV file, and any unsuccessful attempts are recorded separately. The use of random user agents is employed to avoid being blocked by web servers.

2.3. Data Preprocessing

Data preprocessing is a crucial step in the machine learning pipeline, ensuring that the data is clean, consistent, and suitable for the chosen algorithm. In the context of developing a movie recommender system, data preprocessing involves handling missing values, dealing with categorical data, and normalizing numerical data.

2.3.1. Handling Missing Values

Missing values can significantly impact the performance of machine learning algorithms. Therefore, it's essential to identify and address missing values in the dataset. Common strategies for handling missing values include:

1. Imputation:Replacing missing values with estimated values based on statistical methods or patterns in the data.
2. Deletion:Removing instances with missing values, especially if they constitute a small proportion of the dataset.

2.3.2. Dealing with Categorical Data

Categorical data, such as genres, cast, directors, and writers, cannot be directly processed by machine learning algorithms. Therefore, it's necessary to convert categorical data into numerical representations.

2.3.3. Preprocessing Storyline Data

Storyline data requires additional preprocessing steps to extract meaningful features. These steps include:

1. Removing Stop Words:Eliminating common words that don't contribute to the meaning of the text, such as pronouns and prepositions.
2. Tokenizing:Breaking down the text into individual words or phrases.

2.4. Combining features

Combine all the features from the movie into a row,for N number of movies the combination will be a matrix. Matrix construction is a fundamental step in machine learning applications, particularly in recommender systems. It involves representing data in a tabular format, where rows represent entities (e.g., movies) and columns represent features (e.g., genres, keywords, cast, directors, writers, storyline). The resulting matrix serves as the input for various machine learning algorithms, including NMF, which can extract meaningful patterns and relationships from the data. In the context of a movie recommender system, matrix construction involves creating multiple matrices, each representing a specific feature type:

1. Genre Matrix: This matrix represents the presence or absence of various genres in each movie.
2. Keyword Matrix: This matrix represents the occurrence of specific keywords in the storyline or metadata of each movie.
3. Cast Matrix: This matrix captures the popularity of cast members involved in each movie.
4. Director Matrix: This matrix represents the popularity of directors associated with each movie.
5. Writer Matrix: This matrix captures the popularity of writers involved in each movie.
6. Storyline Matrix: This matrix represents the presence or absence of specific words or phrases in the storyline of each movie.

2.5. Matrix Normalization

Matrix normalization is a crucial step in data preprocessing, ensuring that all features contribute equally to the model's training. This is particularly important when dealing with features that have different scales or ranges. Normalization techniques transform the data to have a common range or distribution, preventing features with larger scales from dominating the model's learning process.

In the context of a movie recommender system, normalization is applied to each of the constructed feature matrices. The technique used to normalize is
Min-Max Scaling: This technique transforms the data to lie within a specific range, such as [0, 1].

3. Content based filtering

Content-based filtering (CBF) is a recommender system approach that utilizes the features or characteristics of items (e.g., movies) to recommend similar items to users. CBF algorithms assume that users will prefer items that share similar features with items they have already liked or shown interest in.
In the context of movie recommender systems, CBF involves extracting and representing movie features such as genres, keywords, cast, directors, writers, and storyline. These features are then used to create a similarity matrix that captures the pairwise similarities between movies.

3.1. Cosine Similarity

Cosine similarity is a measure of similarity between two vectors. It calculates the cosine of the angle between two vectors, which represents the directionality of the vectors in a high-dimensional space. A higher cosine similarity value indicates that the two vectors are more aligned, suggesting a closer resemblance between the corresponding items.

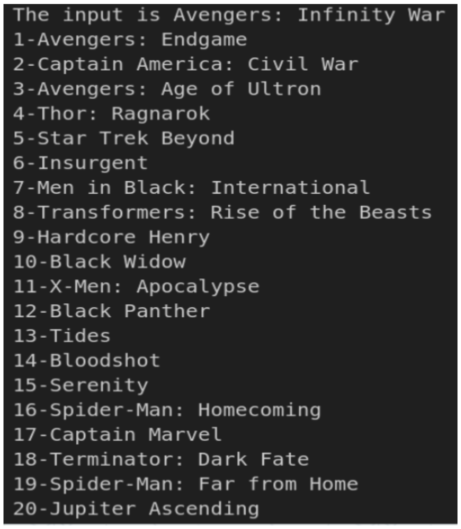
In the context of movie recommender systems, cosine similarity is applied to the movie-feature matrix to compute pairwise similarities between movies. The cosine similarity scores represent the degree to which two movies share similar features and characteristics.

The formula for cosine similarity between two vectors a and b is:
$\cos(a, b) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|}$

a * b is the dot product of vectors a and b
denominator are the magnitudes (lengths) of vectors a and b, respectively
In the context of movie recommendations, higher cosine similarity scores between movies suggest that they share similar features and characteristics, making them likely candidates for recommendation to users who have expressed interest in similar movies.

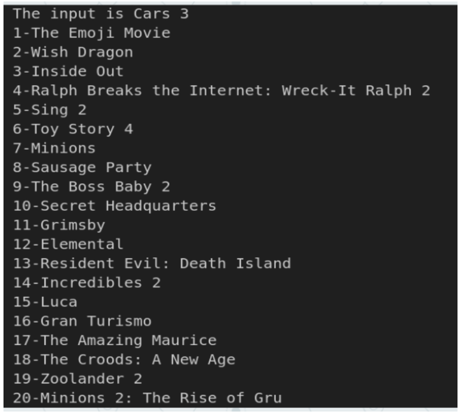
4. Non-negative Matrix Factorization (NMF)

Non-negative Matrix Factorization (NMF) is a dimensionality reduction technique that decomposes a non-negative matrix into two non-negative matrices: a basis matrix and a coefficient matrix. This decomposition enables the extraction of latent features or patterns from the original data



```
The input is Avengers: Infinity War
1-Avengers: Endgame
2-Captain America: Civil War
3-Avengers: Age of Ultron
4-Thor: Ragnarok
5-Star Trek Beyond
6-Insurgent
7-Men in Black: International
8-Transformers: Rise of the Beasts
9-Hardcore Henry
10-Black Widow
11-X-Men: Apocalypse
12-Black Panther
13-Tides
14-Bloodshot
15-Serenity
16-Spider-Man: Homecoming
17-Captain Marvel
18-Terminator: Dark Fate
19-Spider-Man: Far from Home
20-Jupiter Ascending
```

Figure 1. Result 1(content based)



```
The input is Cars 3
1-The Emoji Movie
2-Wish Dragon
3-Inside Out
4-Ralph Breaks the Internet: Wreck-It Ralph 2
5-Sing 2
6-Toy Story 4
7-Minions
8-Sausage Party
9-The Boss Baby 2
10-Secret Headquarters
11-Grimsby
12-Elemental
13-Resident Evil: Death Island
14-Incredibles 2
15-Luca
16-Gran Turismo
17-The Amazing Maurice
18-The Croods: A New Age
19-Zoolander 2
20-Minions 2: The Rise of Gru
```

Figure 2. Result 2(content based)

while preserving its non-negativity.
In the context of movie recommender systems, NMF can be applied to the movie-feature matrix to identify latent features that represent the underlying themes, genres, and characteristics of movies.

4.1. Recombining Normalized Matrices

The normalized matrices constructed for each feature type (genre, keyword, cast, director, writer, storyline) are recombined into a single movie-feature matrix. This matrix represents each movie as a vector of feature weights, where each weight indicates the importance of the corresponding feature in characterizing that movie.
The recombination of normalized matrices allows the NMF algorithm to consider all relevant features simultaneously, enabling it to uncover latent patterns and relationships that span multiple feature categories. This holistic approach leads to more comprehensive and accurate movie recommendations.

The goal of NMF is to approximate the original matrix V as the product of two lower-dimensional non-negative matrices W and H.

V=WH

V is the original non-negative matrix, typically representing data with non-negative values The dimensions of W are m×k, where m is the number of movies and k is the desired number of features.
The dimensions of H are k×n, where n is the number of items.

4.2. Constraint

One of the key characteristics of NMF is the non-negativity constraint, which ensures that all elements in W and H are non-negative.
This constraint has the advantage of providing interpretable and additive components in the factorization.

4.3. Optimization

The factorization is achieved by minimizing the difference between the original matrix V and the reconstructed matrix WH.
This is typically done by defining a cost function, such as the Frobenius norm,
$\sum_{i,j}(V_{ij} - (WH)_{ij})^2$

This objective is minimized during the iterative optimization process of the NMF algorithm. The matrices W and H are adjusted iteratively until the approximation WH closely matches the original matrix V.

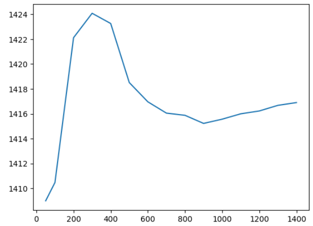


Figure 3. plot for k values (local minimum at 900)

4.4. Reconstruction and Prediction

Once the factorization is complete, the product WH provides an approximation of the original matrix V.
The reconstructed matrix or the Feature matrix can be used for making predictions, such as predicting missing values in the movie-item interaction matrix.

4.5. Challenges

Choosing the right number of features (k) is crucial and often requires hyperparameter tuning.

4.6. Perplexity measure for number of features(K)

Perplexity measure for number of components tells what is the best ‘k’ value we can give for matrix factorization, where ‘k’ is desired number of features. We have calculated uncertainties (Entropy) for all possible values of k, and the one with least uncertainty is the best value for k.

4.6.1. Calculating Entropy For The Matrix

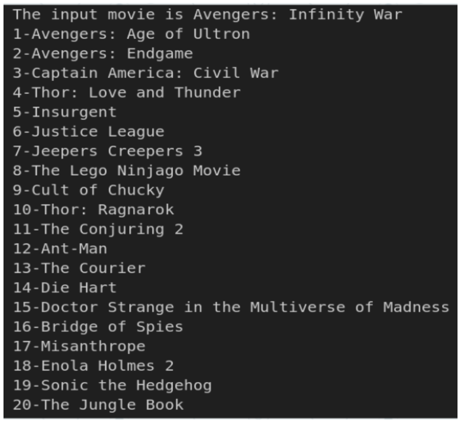
Entropy is the measure of uncertainty. Lower the entropy, better the optimality. So for calculating entropy for each row in the matrix we need to transform our values in the matrix so that these follow the properties of probabilities, that are: All the values in the row are non negative and scaled between

- 0 and 1 (which is done already)

- All the values in a row should sum to 1.

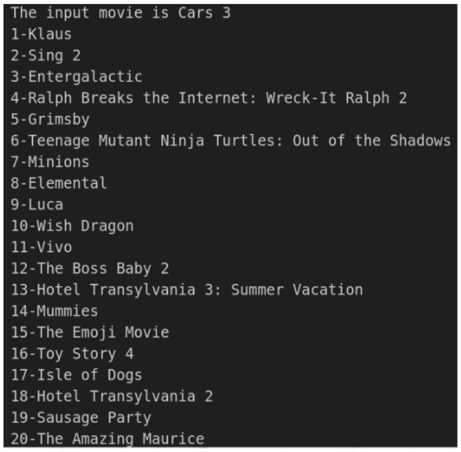
For making all the rows sum to 1, we will divide each row with their corresponding sum of the row.

For finding total perplexity measure, we take mean of all the entropies.
The one with least perplexity measure is the best value. Now, for each row we calculate entropy by:
H(P) = -$\sum_i P(i) \cdot \log_2(P(i))$



```
The input movie is Avengers: Infinity War
1-Avengers: Age of Ultron
2-Avengers: Endgame
3-Captain America: Civil War
4-Thor: Love and Thunder
5-Insurgent
6-Justice League
7-Jeepers Creepers 3
8-The Lego Ninjago Movie
9-Cult of Chucky
10-Thor: Ragnarok
11-The Conjuring 2
12-Ant-Man
13-The Courier
14-Die Hart
15-Doctor Strange in the Multiverse of Madness
16-Bridge of Spies
17-Misanthrope
18-Enola Holmes 2
19-Sonic the Hedgehog
20-The Jungle Book
```

Figure 4. Result 1(Using NMF)



```
The input movie is Cars 3
1-Klaus
2-Sing 2
3-Entergalactic
4-Ralph Breaks the Internet: Wreck-It Ralph 2
5-Grimsby
6-Teenage Mutant Ninja Turtles: Out of the Shadows
7-Minions
8-Elemental
9-Luca
10-Wish Dragon
11-Vivo
12-The Boss Baby 2
13-Hotel Transylvania 3: Summer Vacation
14-Mummies
15-The Emoji Movie
16-Toy Story 4
17-Isle of Dogs
18-Hotel Transylvania 2
19-Sausage Party
20-The Amazing Maurice
```

Figure 5. Result 2(Using NMF)

5. Conclusion

In conclusion, the movie recommendation system has emerged as a powerful amalgamation of various data facets, including genres, cast, directors, and cast and thier popularity rankings The strategic application of Non-negative Matrix Factorization has proven instrumental in distilling latent features that intricately capture user preferences. This, coupled with the refinement introduced by cosine similarity calculations, contributes to the system's ability to offer precise and personalized movie suggestions.

6. Future Scope

The future scope for the movie recommendation system is promising, with opportunities for continuous improvement and innovation. Advanced machine learning techniques, including deep learning, can be explored to enhance the system's predictive capabilities and adaptability to evolving user preferences. Integrating real-time user feedback mechanisms will contribute to dynamic learning and refinement, ensuring the system stays attuned to changing tastes. Hybrid recommendation models, combining collaborative filtering with content-based approaches, offer potential for heightened accuracy, particularly in mitigating the challenges associated with the cold start problem. Embracing emerging technologies such as natural language processing and sentiment analysis can deepen the system's comprehension of user sentiments, providing more nuanced recommendations. Expanding the scope beyond movies to encompass a broader spectrum of entertainment content and platforms will extend the system's impact, establishing it as a versatile and indispensable tool in the realm of personalized content discovery and engagement.

References

@articlesharma2022survey, title=SURVEY ON MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING, author=Sharma, S. and Mahajan, N., journal=International Research Journal of Modernization in Engineering Technology and Science, volume=4, number=12, pages=323–332, year=2022
@INPROCEEDINGS9257290, author=Fanca, Alexandra and Puscasiu, Adela and Gota, Dan-Ioan and Valean, Honoriu, booktitle=2020 21th International Carpathian Control Conference (ICCC), title=Recommendation Systems with Machine Learning, year=2020, volume=, number=, pages=1-6, doi=10.1109/ICCC49264.2020.9257290

@article title=An Introduction to Recommender Systems. In Recommender Systems, author=Aggarwal C.C, journal=Springer: Berlin/Heidelberg, Germany, year=2016
@INPROCEEDINGS7577578, author=Lahitani, Alfirna Rizqi and Permanasari, Adhistya Erna and Setiawan, Noor Akhmad, booktitle=2016 4th International Conference on Cyber and IT Service Management, title=Cosine similarity to determine similarity measure: Study case in online essay assessment, year=2016, volume=, number=, pages=1-6, doi=10.1109/CITSM.2016.7577578
@ARTICLE6165290, author=Wang, Yu-Xiong and Zhang, Yu-Jin, journal=IEEE Transactions on Knowledge and Data Engineering, title=Nonnegative Matrix Factorization: A Comprehensive Review, year=2013, volume=25, number=6, pages=1336-1353, doi=10.1109/TKDE.2012.51

GitHub Link for the project: link