Discovering the Algorithms behind AI-powered Entertainment Platforms

LaTeX template adapted from:
European Conference on Artificial Intelligence

David Akinbode[1]

Abstract. In an ever-expanding technological sphere, engagement has become a new form of currency. Social media platforms, search engines, and streaming services have used AI algorithms to deliver personalized content, keeping users engaged longer. This paper unravels the use of two recommendation systems in the area of movie streaming. The research compares two filtering algorithms, collaborative filtering (using KNN) and content-based filtering, to discover which algorithm is most effective in providing user-specific suggestions. In this study, the MovieLens dataset was used , with 100,000 ratings from 1,000 users on 1,700 movies [4]. The research concluded that collaborative filtering (using KNN) is the most effective choice in recommending movies, with an accuracy score of (MAE: 0.7234) compared to content-based filtering (MAE: 0.8945).

1 Introduction

Recommendation systems exist on various online platforms. From personalized search suggestions to video recommendations on streaming platforms, AI algorithms are the foundation of these processes. The challenge these platforms face is identifying which algorithm will keep users engaged for longer periods of time. With a wide range of user data, different approaches can be used. According to Ozgun and Treske (2021), platforms like Netflix have integrated algorithmic regulation into their program flow, masked behind an interactive menu, which has become a constituent element in how media is distributed [6]. With an ever-growing movie industry, platforms like Netflix, Hulu, and Prime strive to find algorithmic ways to recommend new movies to users based on user preferences.

Two primary approaches to recommendation systems are used in movie streaming platforms: collaborative and content-based filtering. Collaborative filtering uses user profiling to recommend certain movies to users. By using KNN, content is recommended to a user based on that user's top 5 closest neighbours. Collaborative filtering is motivated by the assumption that if two similar users have a similar rating on some items, they will have similar ratings on the remaining items [3]. For example, if a user's neighbours rated a certain action movie highly, then that user would also get recommended that particular movie. Content-based filtering takes a different approach. Content-based filtering looks only at the user and the content and genre of the movie. If a user rates thriller movies highly, then the system will use this data to recommend the user more thriller genre movies. Content-based filtering techniques generate their predictions based on the user information and do not rely on the contributions or ratings from other users [5].

This research uses the MovieLens dataset with 100K entries to compare and analyse the difference between collaborative and content-based filtering when recommending movies to users. Once the dataset is loaded with the proper data (user data, movie data, and user rating metrics), the data is structured and then used to train both models. Once the models are trained, they are evaluated on the test portion of the dataset, and the results are presented.

2 Methodology

2.1 Dataset Description

The MovieLens 100k dataset is comprised of 100,000 movie ratings on a 1-5 scale from 943 users on 1,682 movies. Each user has also rated at least 20 movies, although some users have rated more. From this dataset, 2 files were used to train and test the two models: [u.data] and u.item. [U.data] includes user_id, item_id, rating and timestamp. This data is essential in understanding what users rated what movie, and how well or poorly that movie was rated. The other file, u.item, includes movie data and genre categories that were essential for modelling.

Before being used to train the models, the dataset was split into two sets: 80% for training the models (train_rating) and 20% was used to test the models after training. Once split, the training data is then transformed from a long list into a matrix. This matrix includes the following information:

- index = 'user_id': Each row represents a user
- columns = 'item_id': Each column represents a movie
- values = 'rating': Each cell contains a rating a user gave to that particular movie
- .fillna(0): If a user didn't rate a movie, '0' is used to represent this.

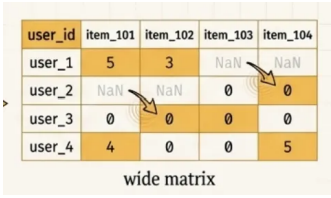

Figure 1. Movie Data Transformed into Matrix

[1] School of Computing and Mathematical Sciences, University of Greenwich, London SE10 9LS, UK, email: da2281j@gre.ac.uk

Figure 1 shows the data that has been transformed into a matrix. Before the data is used to train the model, that data needs to be ***normalized*** (more about this later). Users are categorized based on three factors:

1. Volume: Which is the rating count or the number of movies a user rated (average rating count)
2. Preference: The average rating score of the user - is the user a tough or easy rater?
3. Consistency: Measures the variability of a user's rating pattern. How volatile is their rating?

Once the data is transformed and scaled, it is ready to be used to train the models.

2.2 Algorithm 1: KNN Collaborative Filtering

The process of collaborative filtering works by using KNN. This filtering method is based on user profiling, where a user is suggested a movie based on their closest neighbours. The KNN implementation works like this:

- The training data (matrix) is used to calculate a similarity score (from -1 to 1) between every pair of users based on their rating score vectors. The result is stored in a user-to-user similarity matrix [1].

2.2.1 Understanding Cosine Similarity

To understand this process further, there needs to be an understanding of the deeper mathematical components. The model uses the mathematical function '*cosine_similarity*' to calculate the similarity score between users. Cosine similarity inputs the user's data with movie rating preferences and compares it with another user's data. The angle of the vector that is created is cosined. Figure 2 showcases this process graphed on a 2D plane with the formula below. Cosine similarity between two objects is and compares these objects on a normalized scale.

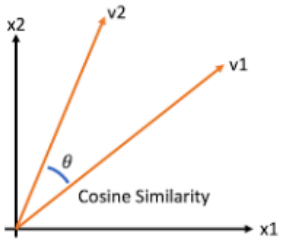

Figure 2. Cosine Similarity Taken From [7]

$$\text{Cosine Similarity} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{1}$$

- The *i* in sigma is an index, and it is used for tracking the movie/user ID we are working on, and *n* represents the movie/user ID we are looking at.
- *A* and *B* represent the data input from the movie/user we are examining.

Once the similarity matrix is calculated, it is used to predict the rating scores of users. Each user is connected to their '*k*' or neighboured users (users with the highest similarity scores), and weights are given to neighbours with higher similarity scores. Predictions of a particular movie are then set based on what the neighbours of that user rated. If a neighbour of that user didn't rate the movie being predicted, this neighbour is filtered out. The final predication is then calculated as a weighted average [2].

2.3 Algorithm 2: Content-Based Filtering

The process of content-based filtering recommends movies based on the content (more specifically, the genre) of the film to users. It works by creating a preference profile based on the movies the user has rated highly. It then recommends movies whose genre makeup is close to the user's profile.

A genre preference is created for a single user by first identifying all movies the user has rated highly (3.5 stars or more). The genre information of that movie is then used to calculate a weighted average for the genre vector. The final vector is normalized to present a user's taste distribution across all genres.

2.3.1 Understanding Normalization

As mentioned earlier, normalization is the process of evening the data. Normalisation is used to ensure that the model doesn't misinterpret the data given. For example, if a user rates more movies than another user, the model could think that the user with more data is more important. Normalisation is used to create data that is even and minimises the risk of misinterpretation. The formula of the StandardScaler function is given below:

$$x_{std}^{(i)} = \frac{x^{(i)} - \mu_x}{\sigma_x} \tag{2}$$

Once the genre preference of a user is generated, the target movie's genre is also obtained. Cosine similarity is calculated between the user's profile and the movie's features. This score measures how well the movie's genre matches the user's taste. Finally, the similarity score is converted into a 1-5 star rating scale.

3 Evaluation Metrics

Once both models are trained, the test data is used to evaluate how well the models perform. Filtering of the testing data is used to ensure that only users with a known movie rating are used. Both collaborative and content-based models make predictions alongside the true actual ratings

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}| \tag{3}$$

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2} \tag{4}$$

The top formula above represents one metric that was used - MAE or mean absolute error. It measures how far the model's prediction was from the actual rating. A MAE of 0.8 represents that the predictions is off by 0.8 stars on average. The formula below it represents the RMSE or root mean squared error. Similar to MAE, RMSE

measures accuracy but squares the error, penalising the model more heavily. For both metrics the lower the score the better the model performed.

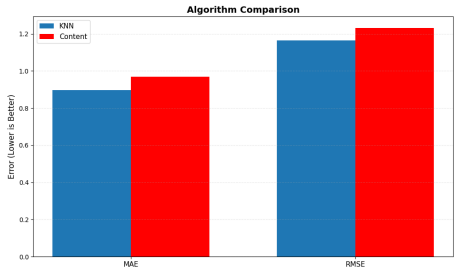

Figure 3. Collaborative and Content-Based Model Error Results

Based on Figure 3, the Collaborative KNN filtering model performs better than the Content-based model. Based on the graph, KNN shows an MAE score of 0.8975 while the content-based filter produced an MAE score of 0.9679. KNN produced a 7.3% improvement in MAE over the content-based model. KNN also produced a RMSE score of 1.1632 while content-based filtering produced a score of 1.2307. This suggests that for the MovieLens dataset, the patterns of what similar users (collaborative) rate are a stronger signal for prediction than the genre information of the movies themselves (content-based)

4 Conclusion and future work

The study presents a comparable analysis of collaborative CNN filtering and content-based filtering on movie and user data taken from the MovieLens 100k dataset. Results showcase that KNN achieved 7.3% better MAE and a better (lower) RMSE score when compared to content-based filtering.

Future work would involve using other MovieLens dataset information, such as user demographic information, and more movie content information, such as actor list, duration of movie, to determine user preference. Future work can also use other algorithms to compare user movie preferences and a more recent movie dataset.

Artificial intelligence is integrated into various social media and entertainment platforms. In order for streaming platforms such as Netflix, Prime, and YouTube to have an improved recommendation system, collaborative KNN-based AI methods are proven more effective over content-based methods.

ACKNOWLEDGEMENTS

This coursework builds on foundational recommendation systems research pulled from [2]. Implementation used Python with scikit-learn, pandas, and NumPy libraries. Data courtesy of GroupLens taken from Kaggle.

REFERENCES

[1] Ario Bagus Bramantyo and Agung Toto Wibowo, 'Collaborative filtering movie recommendation system using k-means clustering with particle swarm optimization', *https://doi.org/10.1109/icodsa58501.2023.10276506.*, (2023).

[2] Fan Y.-C. Ji Y. Zhang J. and Sun A., 'Our model achieves excellent performance on movielens: What does it mean? acm transactions on information systems', *https://doi.org/10.1145/3675163.*, (2024).

[3] Lee J. Sun M. and Lebanon G, 'A comparative study of collaborative filtering algorithms', *https://doi.org/10.48550/arxiv.1205.3193.*, (2012).

[4] MovieLens, 'Movielens 100k dataset', *https://www.kaggle.com/datasets/prajitdatta/movielens-100k-dataset.*, (2016).

[5] Pradeep N. Rao Mangalore K.K. Rajpal B. Prasad N. and Shastri R., 'Content based movie recommendation system', *https://doi.org/10.22105/riej.2020.259302.1156.*, (2020).

[6] Aras Ozgun and Andreas Treske, 'On streaming-media platforms, their audiences, and public life', *https://doi.org/10.1080/08935696.2021.1893090*, (2021).

[7] T. Narula T. Singh R.H. Maurya S. Tripathi and Srivastav G, 'Movie recommendation system using cosine similarity and knn', *https://doi.org/10.35940/ijeat.e9666.069520.*, (2020).