UNIVERSITY OF THESSALY, VOLOS
DEPT. OF ELECTRICAL AND COMPUTER ENGINEERING

ECE461 - Machine Learning for Data Science and Analytics

Movie Recommendation System

Efthymia Koustika 3172
Aristea Koutsomarkou 3370

FALL SEMESTER, JANUARY 2024

Contents

1 Abstract

This report introduces the development of a movie recommender system aimed at enhancing user engagement and satisfaction in the vast landscape of cinematic options. Due to the variety of movie options and the growth of digital streaming services, people frequently experience decision fatigue and find it difficult to choose and discover movies that align their preferences. The motivation behind this project is to address this issue by developing a personalized movie recommendation system that takes user preferences, past viewing behavior, and film features into account.

The methodology employed in this project includes collaborative filtering (matrix factorization, knn and neural networks),content-based filtering, and a hybrid approach that combines collaborative and content-based filtering. Content-based filtering uses item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback. To address some of the limitations of content-based filtering, collaborative filtering uses similarities between users and items simultaneously to provide recommendations. There are two classes of Collaborative Filtering: a) User-based, which measures the similarity between target users and other users and b) Item-based, which measures the similarity between the items that target users rate or interact with and other items.

2 Introduction

The increasing amount of digital content available, especially when it comes to movies, makes it very difficult for customers to choose content that suits their tastes. Due to the large number of streaming services and a rapidly expanding movie library, consumers frequently face a dilemma of choice, which results in decision fatigue and less-than-ideal user experiences. There has never been a greater need for an effective and customized movie recommendation system.

The significance of resolving this issue is that it will improve user satisfaction as well as engagement in the wide range of movies that are accessible. A well-designed recommendation system helps users find information more easily and makes decisions easier for them. This increases platform loyalty and user retention. The capacity to provide personalized recommendations has a strong association with both business performance and customer satisfaction measures in the highly competitive entertainment services market.

The motivation to improve user experience and give everyone a more unique and entertaining cinematic journey is the reason behind this problem's pursuit. We are interested in changing the way consumers interact with with movie platforms by using algorithms and machine learning approaches in order to make sure they constantly receive material that fits their personal preferences and viewing history.

In general our recommendation algorithms analyze the user interaction data including historical movie preferences and viewing habits in order to to generate personalized movie recommendations. The input varies from movie or user id to movie title depending on each algorithm. We then use each of our implemented algorithms (knn , matrix factorization, neural network, content-based,and hybrid approach) to get a list of recommended movies (using the predicted ratings that were calculated by the algorithms).

3 Literature Review

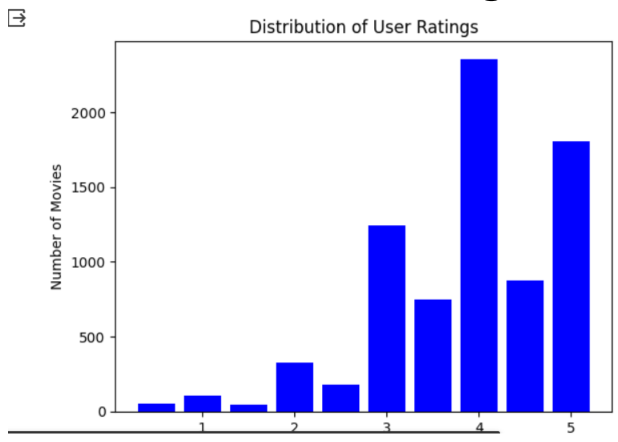
In this paper[1], the method implemented involves web scraping movie posters from the TMDB website using the Python library urllib and requests. The recommender system utilizes ConvNets for finding similar movie posters based on visual aspects. The system uses a pre-trained VGG16 model on ImageNet to save time and leverage state-of-the-art model capabilities. The features learned by the model are used for meaningful recommendations, and the output layer is removed to treat the rest of the ConvNet as a feature extractor for movie posters. The system also makes use of A/B testing to compare recommendation approaches. No clear evaluation metric was defined, and the ConvNet features performed better than using only the raw image array as an input. Another paper[19] discusses the implementation of matrix factorization as a method for collaborative filtering in recommender systems. It describes the use of factor vectors to estimate user-item ratings and the minimization of the regularized squared error to learn the factor vectors. The paper also discusses the use of stochastic gradient descent and alternating least squares as learning algorithms for minimizing the error function. Additionally, it addresses the incorporation of biases to capture user and item effects in the rating values. The writer of this paper[12] uses a combination of demographic, content-based, collaborative, social-based, context-aware, and hybrid filtering algorithms to build a recommender system. They discuss the cold-start problem, kNN algorithm, similarity measures, and the quality of these measures. The proposed method in this paper[6] for predicting user preferences in recommendation systems is a hybrid approach based on naïve Bayesian classifier with Gaussian correction and feature engineering. It consists of four stages: Preprocessing, Feature Engineering, Naïve Bayesian Model, and Prediction. The process involves data cleaning, feature selection, model creation, and prediction using the Movie Lens 100k dataset. One last interesting paper[24] explores various collaborative filtering approaches in recommender systems, including addressing cold-start recommendations, probabilistic model estimation, structural extension to logistic regression, associative retrieval techniques, maximum-margin matrix factorization, imputation methods, SVD-based algorithms, and cluster-based smoothing.

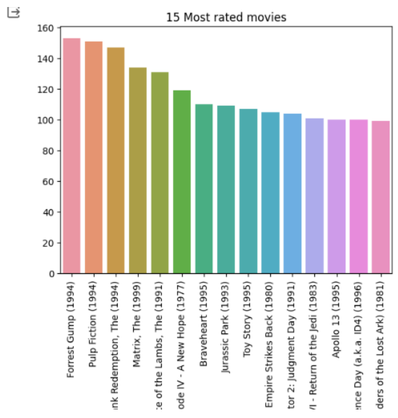
4 Dataset and Features

For the purposes of the proposal and implementation of our recommender system, we selected the MovieLens dataset (ml-latest-small), which is a database of personalized ratings of various movies from a large number of users. The dataset describes 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service. It contains 100836 ratings and 3683 tag applications across 9742 movies. These data were created by 610 users between March 29, 1996 and September 24, 2018. All selected users had rated at least 20 movies. No demographic information is included. Each user is represented by an id, and no other information is provided. The data are contained in the files ‘links.csv‘, ‘movies.csv‘, ‘ratings.csv‘ and ‘tags.csv‘.

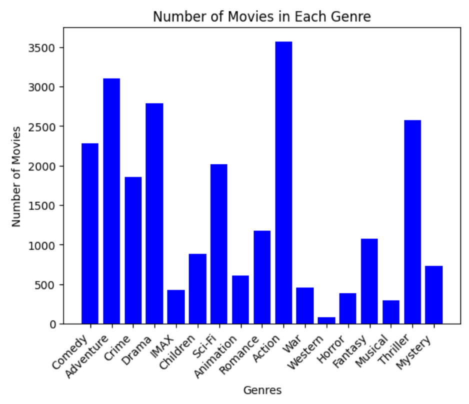
MovieLens users were selected at random for inclusion. Their ids have been anonymized. Movie and user ids are consistent between ‘ratings.csv‘, ‘tags.csv‘, ‘movies.csv‘, and ‘links.csv‘. The preparation of data is a critical step that significantly influences the performance of models. We checked the data for duplicates and missing values. These would affect dramatically our results. In our case we didn’t have any so we proceeded to further analysis. To gain insights into the dataset, we performed exploratory data analysis (EDA). Key aspects of our analysis include:

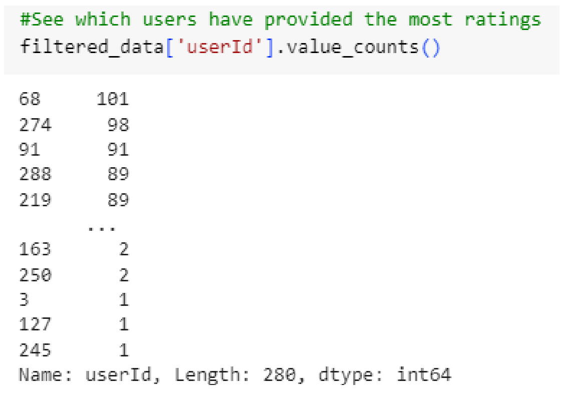
Rating Distribution





Genre Analysis and user activity





```
#See which users have provided the most ratings
filtered_data['userId'].value_counts()
```

```
68     101
274     98
91      91
288     89
219     89
      ...
163      2
250      2
3        1
127      1
245      1
Name: userId, Length: 280, dtype: int64
```

5 Methods

1. **Collaborative Filtering (CF)** is a powerful and widely used approach in recommendation systems that relies on the ratings and the collective preferences of users. Singular Value Decomposition (SVD) is one of the common matrix factorization methods used in collaborative filtering. SVD decomposes the user-item matrix into three matrices U, S and V^T.

$$\mathbf{X} = \mathbf{U}\boldsymbol{S}\mathbf{V}^T \tag{1}$$

where U represents users, S is a diagonal matrix with singular values capturing latent factors, and V^T represents items. The singular values and vectors capture latent factors, representing hidden features that reveal underlying patterns in user-item interactions.

1.1 User-based method recommends products to a user based on the fact that the products have been liked by users similar to the user.

1.2 Item-based is a recommendation technique that suggests items to a user based on the similarity between items they have shown interest in or interacted with in the past.

2. **k-Nearest Neighbors** is a collaborative filtering algorithm that utilizes distance measurements between samples and other data points in a dataset to predict ratings. It identifies the K-nearest neighbors and utilizes majority voting to make rating predictions.

3. **Cosine Similarity** measures the difference between two non-zero vectors in an inner product space. Mathematically, it computes the cosine of the angle between two vectors projected in a multidimensional space. The smaller the angle, the greater the cosine similarity. It measures how similar the preferences of two users are. We use it for both user-to-user recommendations and identifying related items (movies).

4. **Content Based filtering** recommends movies by analyzing the characteristics of movies (genre, actors, directors) and the user's historical preferences. In our case **TF-IDF** method is used for assessing token importance in text, applied to find similar movies using genre and title. It involves transforming texts into real-valued vectors through the product of Term Frequency (TF) and Inverse Document Frequency (IDF), enabling cosine similarity comparisons.

5. **Neural Network** employs matrix factorization through embeddings to capture latent factors for both users and movies. It takes user and movie IDs as input, embedding them into lower-dimensional vectors, and computes the dot product of these embeddings to predict movie ratings.

6. **Hybrid approach** combines collaborative filtering and content-based filtering. Features from both collaborative filtering and content-based filtering are used. The recommendations generated by each method are assigned weights, and the final recommendation is a weighted combination of the individual recommendations.

6 Analysis

6.1 Experiments

SVD was applied for collaborative filtering. We used the Surprise library's SVD algorithm. The algorithm was instantiated with default **hyperparameters**, including 100 latent factors, 20 epochs, 0.005 learning rate for SGD, and 0.2 regularization term for all parameters. Additionally we experimented with **Cross validation** with 5 and 10 folds

We build our **Neural network** model using Keras with two embedding layers (for users and movies) and a dot product layer to capture interactions. The embeddings are used to represent users and movies in a lower-dimensional space. In this method we primarily experimented with different values of the n_factors and learning rate in order to assess the best results for our model. Choosing the learning rate and n_factors involves trying different values to find the ones that make the optimization process effective and stable.

While implementing the **KNN** model, the cosine similarity metric was chosen because it is commonly used in recommendation systems. It is particularly useful when dealing with sparse datasets, like ours, and helps in providing meaningful recommendations. As for the algorithm parameter, we opted for the 'brute' one. The brute-force approach is straightforward and efficient. It computes distances between all pairs of points, making it suitable for smaller datasets. The chosen hyperparameters were those that consistently yielded satisfactory results in terms of accuracy and computational efficiency.

Regarding the **Hybrid approach** that combines the collaborative and content-based filtering the choice of the weights was considered important. We tried out many different combinations of weights to find a balance between the strengths of collaborative and content-based filtering for our dataset.

6.2 Results

The primary metric that we used is the Root Mean Square Error **(RMSE)**. It is a commonly used metric in recommendation systems. RMSE measures the average magnitude of the errors between predicted and actual ratings. The square root of the mean squared error (RMSE) has the same unit as the target variable (movie ratings in this case), making it interpretable and easy to communicate.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{2}$$

The following table presents the Root Mean Square Error (RMSE) values corresponding to each method

Method	SVD(CF)	KNN	Content-Based	Hybrid	Neural Network
RMSE	0.8306	0.6598	0.934	0.96	1.11

Table 1: Method Evaluation

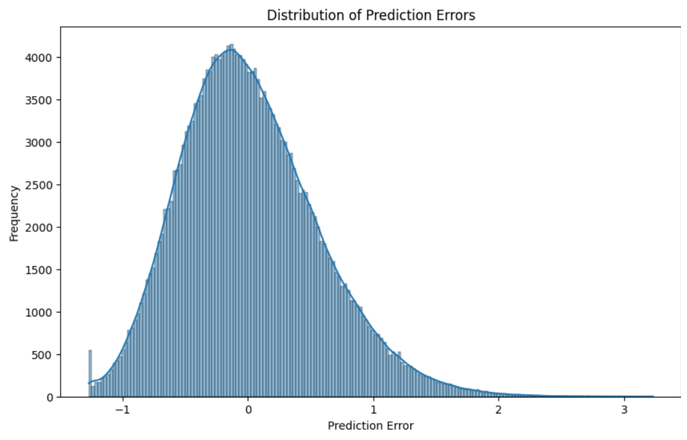
After we applied Cross-Validation for SVD evaluation , we obtained the following results

	5-folds	10-folds
RMSE	0.8375	0.8331

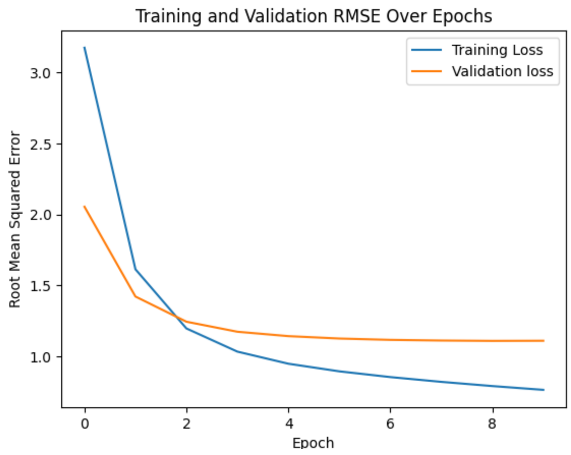
Table 2: RMSE-Cross Validation

6.3 Discussion

In this section, we delve into the outcomes derived from the implementation of the methods we used for movie recommendation. In the figure 1 above we can see the error distribution of SVD **(a)** which suggest that the prediction model is generally accurate because he most frequent errors are

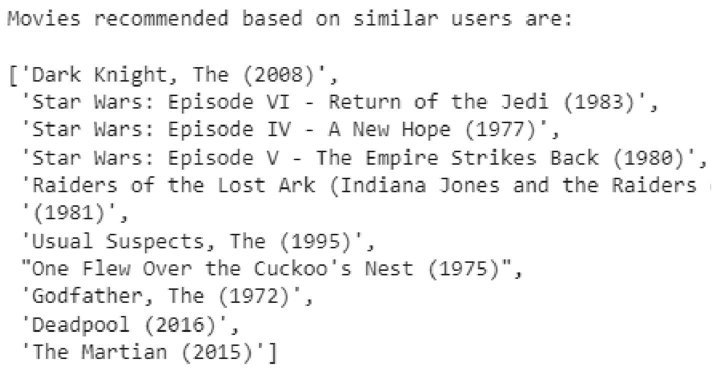


(a)



(b)

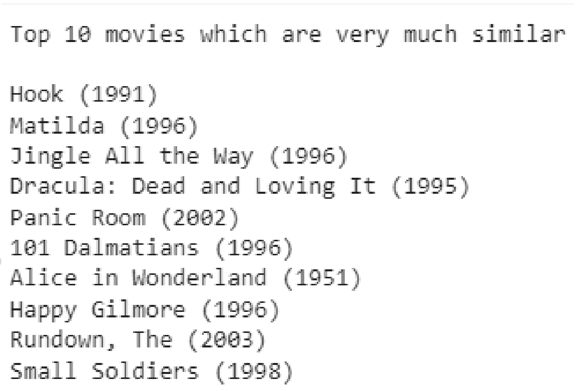
Figure 1: (a)Error distribution SVD and (b)Loss in Neural Network



```
Movies recommended based on similar users are:

['Dark Knight, The (2008)',
 'Star Wars: Episode VI - Return of the Jedi (1983)',
 'Star Wars: Episode IV - A New Hope (1977)',
 'Star Wars: Episode V - The Empire Strikes Back (1980)',
 'Raiders of the Lost Ark (Indiana Jones and the Raiders
 '(1981)',
 'Usual Suspects, The (1995)',
 "One Flew Over the Cuckoo's Nest (1975)",
 'Godfather, The (1972)',
 'Deadpool (2016)',
 'The Martian (2015)']
```

(a) Recommended movies on user 67



```
Top 10 movies which are very much similar

Hook (1991)
Matilda (1996)
Jingle All the Way (1996)
Dracula: Dead and Loving It (1995)
Panic Room (2002)
101 Dalmatians (1996)
Alice in Wonderland (1951)
Happy Gilmore (1996)
Rundown, The (2003)
Small Soldiers (1998)
```

(b) Recommended movies for Space Jam

Figure 2: (a) user-based and (b) item-based

small. We also see the training and validation RMSE over the epochs in the neural network model **(b)**, and both of them get lower over the epochs.

Each method was evaluated by the RMSE as shown in the tables 1 and 2. Overall the performance of the implemented methods was adequate, considering the rating scale, which ranged from 0.5 to 5. Lower RMSE values indicate better performance. In our case KNN and SVD have the lowest values. The SVD Collaborative Filtering method demonstrated a reasonable performance with an RMSE of 0.8306. This was expected because SCD performs well on sparse data. The KNN approach outperformed SVD with an RMSE of 0.6598, indicating its effectiveness in capturing local user preferences. However this might not be the case, because this was a memory-based approach (K-Nearest Neighbors) which means that it performs good in this specific dataset. We then used the KNNBasic from the surprise library, which is a model based approach, and after cross validation we got a higher RMSE value (0.9757). This is because the algorithm learns a model to predict user-item ratings.

A hybrid approach in recommendation systems is the combination of different techniques that enhances the overall recommendation performance and user satisfaction. The RMSE of this method is also quite good but not better than SVD and KNN. This could be due to limitations on the availability and quality of item features.

7 Conclusion and Future Work

Conclusion

In conclusion, the goal of this project was to create a movie recommender system in order to reduce the strain of having to choose a movie. The methodology involved leveraging various recommendation techniques, including collaborative filtering (SVD, KNN, neural networks), content-based filtering, and a hybrid approach, to offer personalized movie suggestions based on user preferences and movies features. The MovieLens dataset was used. Root Mean Square Error highlighted the effectiveness of SVD and KNN in achieving accurate predictions. The hybrid approach, combining collaborative and content-based filtering, also showed promising results.

Future Work

Future work in movie recommendation systems could involve integrating advanced machine learning techniques, particularly deep learning architectures, to enhance predictive accuracy and feature representation. Exploring the inclusion of contextual information, like user demographics and temporal dynamics, may further enhance personalization. Addressing the cold start problem for new users/items and exploring alternative evaluation metrics, such as diversity and user satisfaction, are crucial avenues for research.

8 Contribution

In the development of our project, the Movie Recommendation System, each team member actively contributed in various aspects of the project, ensuring a well-rounded and comprehensive outcome. Our teamwork was characterized by open communication, and a mutual understanding of our respective strengths.

9 References and Bibliography

References

[1] Manan Bhatia Ajay Kaushik, Shubham Gupta. A movie recommendation system using neural networks. *International Journal of Advance Research, Ideas and Innovations in Technology*, 2018. URL https://scholar.google.gr/scholar?hl=en&as_sdt=0%2C5&as_vis=1&q=A+Movie+Recommendation+System+using+Neural+Networks+&btnG=.

[2] Djellal Mohamed Aniss. Recommender system for movie ratings dataset. https://www.kaggle.com/code/djellalmohamedaniss/recommender-system-for-movie-ratings-dataset, 2021.

[3] S. K Singh K. K. Shukla Anuranjan Kumar, Sahil Gupta. Comparison of various metrics used in collaborative filtering for recommendation sys-

tem. https://ieeexplore.ieee.org/abstract/document/7346670, 2015.

[4] Bhawna. Movie recommendation system with neural networks and collaborative filtering. https://medium.com/@bhawna7374/movie-recommendation-system-with-neural-networks-and-collaborative-fil 2020.

[5] Emine Cerit. Movie recommender matrix factorization based. https://www.kaggle.com/code/eminecerit/movie-recommender-matrix-factorization-based, 2022.

[6] Antonio Hernando Jian Wei Kebin Wang Chun-Liang Li, Jianjun Xie and Ying Tan. A hybrid approach for movie recommendation system using feature enngineering. 2018. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8473335.

[7] Codedamn. Machine learning in recommendations systems: An overview. https://codedamn.com/news/machine-learning/machine-learning-in-recommendations-systems-an-overview, 2023.

[8] Heeral Dedhia. Movie ratings and recommendation using knn. https://www.kaggle.com/code/heeraldedhia/movie-ratings-and-recommendation-using-knn, 2021.

[9] GeeksforGeeks. Python implementation of movie recommender system. https://www.geeksforgeeks.org/python-implementation-of-movie-recommender-system/, 2023.

[10] GeeksforGeeks. Recommendation system in python. https://www.geeksforgeeks.org/recommendation-system-in-python/, 2023.

[11] Ibtesama. Getting started with a movie recommendation system. https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system, 2020.

[12] A. Hernando J. Bobadilla, F. Ortega and A. Gutiérrez. Knowledge-based systems. *Universidad Politécnica de Madrid*, 2013. URL https://www.sciencedirect.com/science/article/abs/pii/S0950705113001044.

[13] jhihan. Hybrid recommendation system. https://github.com/jhihan/Hybrid-Recommendation-System/blob/master/Hybrid_Recommendation_System.ipynb, 2020.

[14] jowoojun. Collaborative filtering keras. https://github.com/jowoojun/collaborative_filtering_keras/blob/master/Recipe_Training.ipynb, 2018.

[15] Vibhu Mishra. Movielens recommender system. https://www.kaggle.com/code/vibhumishra707/movielens-recommender-system, 2023.

[16] Kartik Parsoya. Movie recommendation system using knn. https://www.kaggle.com/code/kartikparsoya/movie-recommendation-system-using-knn, 2021.

[17] Prashant. Recommender systems in python. https://www.kaggle.com/code/prashant111/recommender-systems-in-python#8.-Hybrid-Recommender-Systems-, 2021.

[18] Alyssa Q. Simple movie recommender using svd. https://alyssaq.github.io/2015/20150426-simple-movie-recommender-using-svd/, 2015.

[19] Chris Volinsky Robert Bell and Yehuda Koren. Matrix factorization techniques for recommender systems. *IEEE Computer So-*

ciety, 2009. URL https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix].pdf.

[20] rposhala. Recommender system on movielens dataset. https://github.com/rposhala/Recommender-System-on-MovieLens-dataset/blob/main/Item_based_Collaborative_Recommender_System_using_KNN.ipynb, 2020.

[21] rposhala. Movie recommendation system using knn algorithm. https://github.com/rposhala/Recommender-System-on-MovieLens-dataset/blob/main/Item_based_Collaborative_Recommender_System_using_KNN.ipynb, 2020.

[22] Sharmin2697. Movie recommender system. https://github.com/sharmin2697/Movie-Recommender-System/blob/main/code/Functions.ipynb, 2021.

[23] SJD1882. Big data recommender systems. https://github.com/SJD1882/Big-Data-Recommender-Systems/blob/master/notebooks/MovieLens27M-ALS-Recommender-System.ipynb, 2019.

[24] Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Department of Computer Science and Engineering, Florida Atlantic University,*, 2009. URL https://www.hindawi.com/journals/aai/2009/421425/.

[25] Jalaj Thanaki. Movie recommendation engine. https://github.com/jalajthanaki/Movie_recommendation_engine/blob/master/Movie_recommendation_engine.ipynb, 2018.

[26] M. Chandrashekhar Anirudh Challa V. Subramaniyaswamy, R. Logesh

and V. Vijayakumar. A personalised movie recommendation system based on collaborative filtering. https://www.inderscienceonline.com/doi/abs/10.1504/IJHPCN.2017.083199, 2017.

[27] Surhan Zahid. Recommendation system using matrix factorization. https://github.com/SurhanZahid/Recommendation-System-Using-Matrix-Factorization, 2018.