Movie Recommendation System

Jhansi Saketa B V  
 MS in Computer Science  
University of Colorado  
 Boulder, CO  
 jhbh9073@colorado.edu

Danielle Aras  
 BS in Computer Science  
 University of Colorado  
 Boulder, CO  
 danielle.aras@colorado.edu

Soumyadeb Maity  
 MS in Computer Science  
 University of Colorado  
 Boulder, CO  
 soma5722@colorado.edu

MOTIVATION

Movies are one of the top all-time sources of entertainment. People enjoy watching, reviewing, and discussing them. For many people reviewing and rating movies is a passion. A movie can have fans and critics too. So, in a crowded entertainment market, movie streaming services and advertisers need to present viewers with the most relevant movies possible to maintain customer interest and loyalty.

Many companies now use data mining algorithms to create a recommendation system to match users with movies they would likely enjoy. A movie recommendation system can leverage data collected from users to identify patterns in an individual or group's viewing history, then use this insight to identify movies to recommend to a specific viewer. This project will use a database of user-submitted movie ratings to explore ways to generate movie recommendations and predict how users may rate future movies. Multiple algorithms will be used in order to compare their performance and practicality.

LITERATURE SURVEY

One of the better-known recommender algorithms is the Netflix Recommender System [1]. The algorithm originally began as a simple predictor of how many stars a user would rate a movie on a scale of one to five based on previous ratings but has grown into a multi-faceted system that generates many different types of recommendations according to other types of data Netflix now collects. For example, it can recommend titles that are popular among the entire user base, or movies that are similar to a single movie a viewer has watched based on a similarity algorithm.

Another system, MovieGEN, relies more on direct user input by asking viewers to respond to a list of questions and then generate recommendations based on their answers [2]. It uses both clustering and SVD to generate recommendations. Other systems use clustering to identify groups of users with similar tastes [3] or TF-IDF to find similar movies [4].

PROPOSED WORK

We will use movie rating data from the MovieLens dataset. This dataset is almost clean and can be used directly with some minimal pre-processing. We will perform basic cleaning to check for null values (movies without genres, missing ratings, invalid ratings). We will also combine the separate csv files for ratings and movies into a single table so ratings are linked to movie genres. Lastly, we will create a user genre rating table with average ratings for each genre of movie for each user.

This Project “Movie Recommendation System” generally will use two major techniques:

1. Content-based filtering uses algorithms which will identify similar items to items the user already likes i.e., items that have comparable properties independent of other users' data. A common algorithm for content-based filtering is “Term Frequency-Inverse Document Frequency" or TF-IDF. This is the most used algorithm to convert text into vectors and is widely used to extract features across various NLP applications but can also be applied to a recommender system. Another common algorithm is dot product / sum of product. For our movie recommendation system, content-based filtering will be used to identify movies that are similar to those that viewers have rated highly based on their genres.
2. Collaborative-based filtering uses data from other users to make recommendations to a given user. Methods include clustering, which groups similar users together and uses the top-rated movies from a user's cluster as recommendations, and classification algorithms like KNN and SVD.

The problem with a collaborative filtering is the cold start problem, once the model is deployed, the main concern is that the system cannot draw any inferences for movies or users about which it has not yet gathered sufficient information.

We will explore both content-based and collaborative-based filtering methods and seek to create an optimal movie recommendation system using techniques from previous recommender systems. While the Netflix recommendation system is highly developed and combines many different techniques, for our purposes and with the limited features in our dataset we will focus on simpler algorithms. We also do not have the ability to use an interactive process like MovieGEN’s question system as we do not have direct access to the millions of users in our dataset. Therefore, we will focus on collaborative-based filtering using clustering and classification, and content-based filtering using TF-IDF and dot product.

We will use K-means clustering to divide reviewers into clusters with similar taste in movies based on their average ratings of different genres. Clusters can be used to predict a user’s rating for a movie they have not seen yet based on the cluster average rating. “Top” movies for each cluster can be found by identifying the highest-rated movies for each cluster. Different k values will be used to find the optimum number of clusters.

DATA SET

The Dataset that we are using for this project is from MovieLens Dataset. There are different sizes of the dataset provided, so we will use the “Latest Full” dataset for the project which has 27 million data points. The “Latest Small” dataset can be used for testing the model which has nearly 100,000 data points. The data can be accessed in the URL: <https://grouplens.org/datasets/movielens/> with the permission of the MovieLens organization. We completed their contact form and successfully got permission to use the dataset in our project.

EVALUATION METHODS

1. For clustering, the fit of the clusters will be evaluated by the sum of the squared error (SSE). The lower the SSE, then the more similar are the Users/Movies. A high SSE suggests that the Movies/Users in the same similar segment have a reasonable degree of differences between them and may not be a true (or usable) segment. We will also look at the silhouette coefficient to judge how well-separated our clusters are. The silhouette coefficient ranges from 1 to –1. Positive values mean clusters are more separate, while negative values mean there may be confusion or incorrect assignments to clusters.
2. The optimum number of clusters will be chosen based on the elbow method and silhouette coefficient.
3. We will use the inbuilt accuracy methods of the KNN and SVD algorithms in sklearn.

TOOLS

The following tools are going to be used to develop the project.

1. Python

Python programming language is the core tool to build our project. The data extraction, data cleaning, pre-processing and all the models and algorithms will be built using python programming language.

1. Pandas

We will use the Pandas Python module to work with the dataset. The major role of Pandas in our project is extraction of the data, creating data frames, and working with the data frames.

1. Scikit

Scikit is a Python library that contains various classification, regression, and clustering algorithms. We will use the clustering, KNN, SVD, and TF-IDF modules in particular.

1. Matplotlib

Matplotlib is a plotting library for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) programming language and its numerical mathematics extension [NumPy](https://en.wikipedia.org/wiki/NumPy). In our project, we use matplotlib to plot the graphs or visualize the results.

MILESTONES

Week 10: Complete preprocessing and initial visualizations. Begin implementation of planned algorithms. Begin testing on reduced test dataset and refining/adjusting algorithms based on evaluation methods.

Week 11: Apply algorithms to full dataset and evaluate results. Final adjustments and add all code to Github. Complete project progress report and begin final report.

Week 12: Create final visualizations of results. Finish project report and presentation. Add all material to Github and submit.

MILESTONES COMPLETED

1. Exploratory Data Analysis
   1. knowing how many distinct users and movies are in the dataset.
   2. Worked on following features
      1. Movies with most views.
      2. Users with most views.
2. Clustering Implementation
   1. Completed preprocessing to use data in k-means clustering, including combining review and movie CSV files into new Pandas dataframe and grouping reviews by user and genre
      1. Null values (genres with no reviews by a user) were replaced by 0 as the absence of any reviews suggests a dislike or lack of interest in the genre. This was tested against replacing with 2.5 (neutral like/dislike value), which resulted in less distinct clusters.
   2. Tested clustering algorithm on reduced dataset to adjust parameters and select features (genres) to use
      1. Clusters evaluated using the elbow method and silhouette coefficient
      2. As elbow method resulted in graphs with a relatively smooth curve and no noticeable “elbow”, we will rely more on the silhouette coefficient for model selection
   3. Began applying algorithm to full dataset and making adjustments
   4. Began writing methods to get different types recommendations based on clusters (top reviewed movies, most watched, top in genres, etc.)
3. Began KNN implementation

MILESTONES TODO

1. Continue the implementation of KNN, and Clustering
   1. Finish adjusting k-means clustering algorithm to full dataset and complete recommendation functions
2. Implement SVD
3. Visualize the results

RESULTS SO FAR

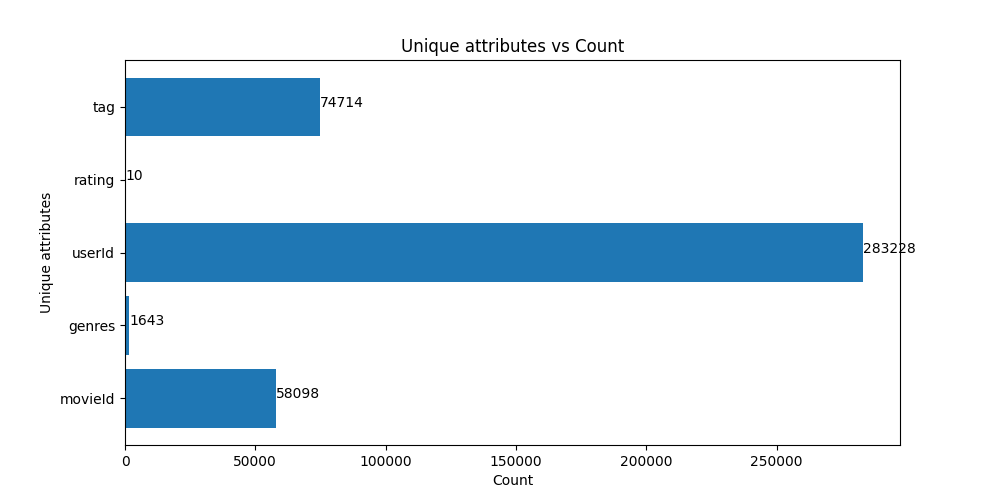


FIGURE 1

Figure 1 shows the Distinct count of each unique attribute like, number of tags, rating, userID, genres, movies. The Dataset used Latest Full 27 Million data points.

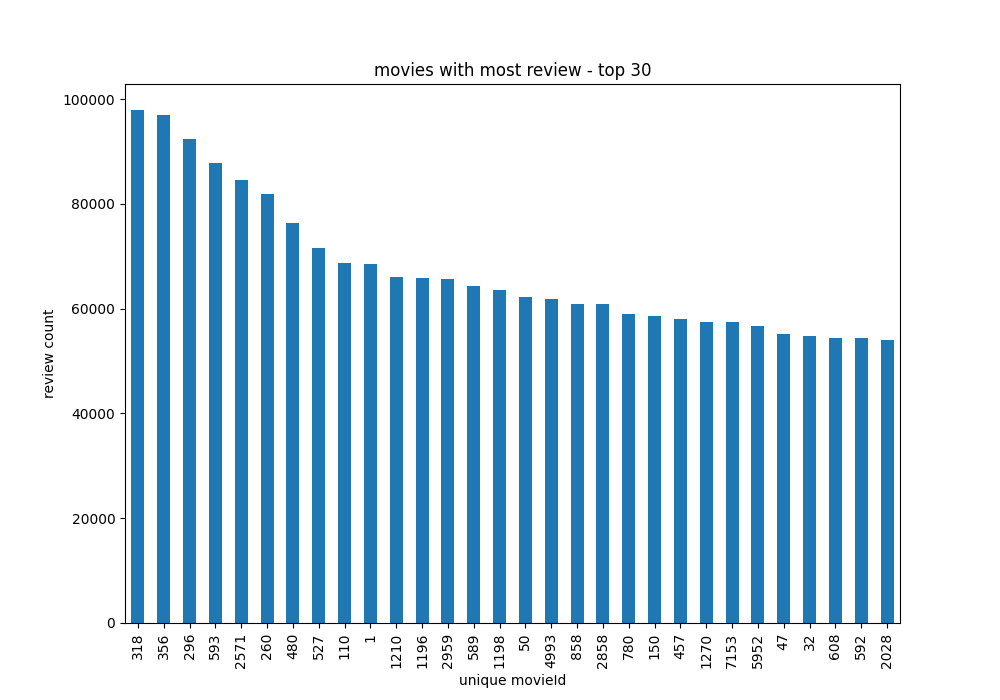


FIGURE 2

Figure 2 is a histogram of the top 30 most-reviewed movies from the full dataset.

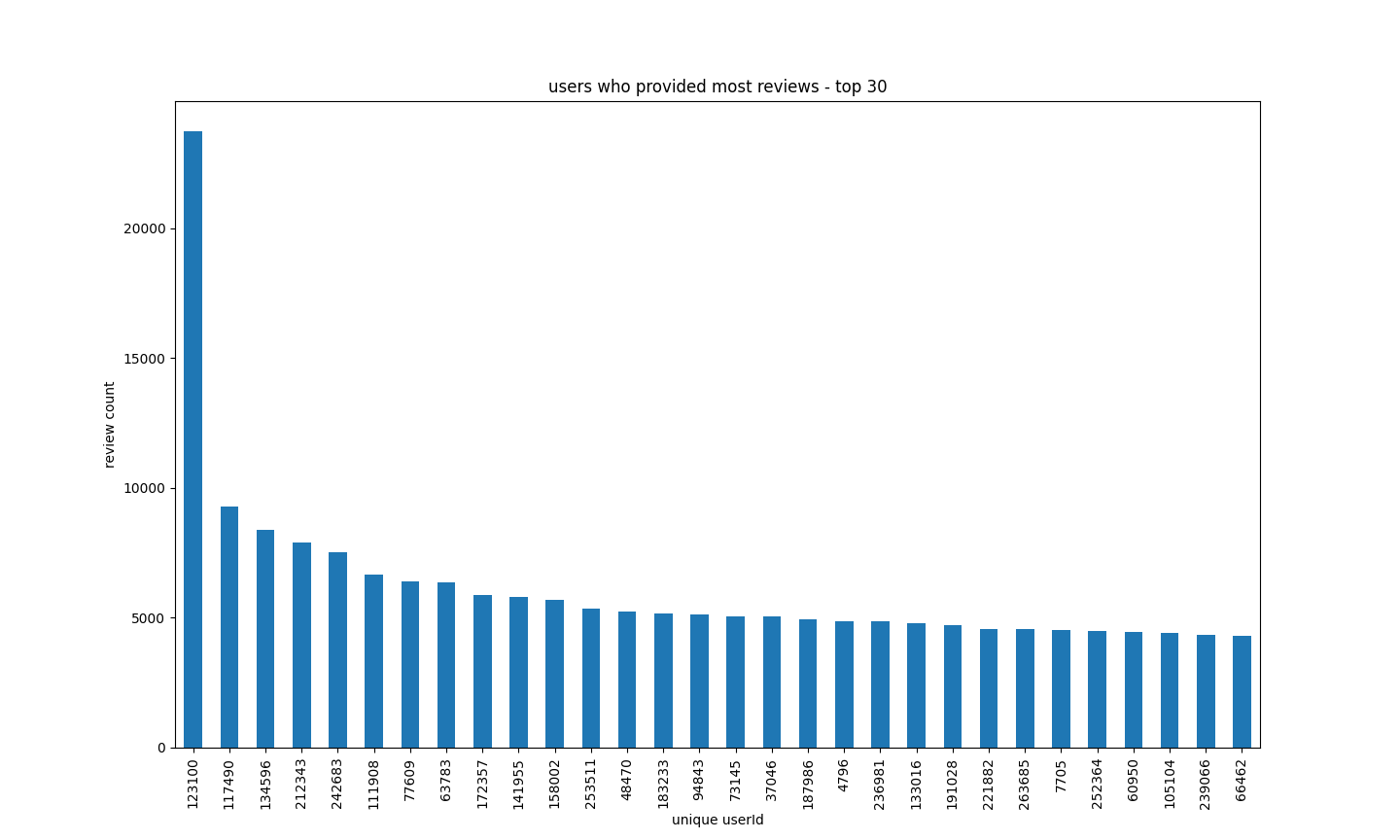


FIGURE 3

Figure 3 is a histogram of the top 30 users with the most reviews

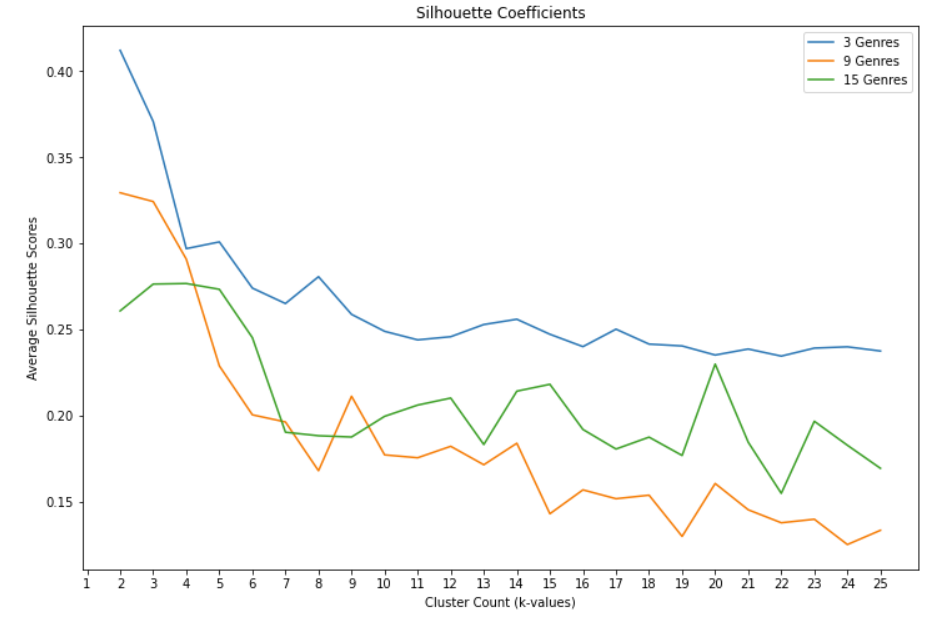


FIGURE 4

Figure 4 shows a plot of the silhouette coefficients of k-means clustering using 3, 9, and 15 genres based on the reduced dataset.

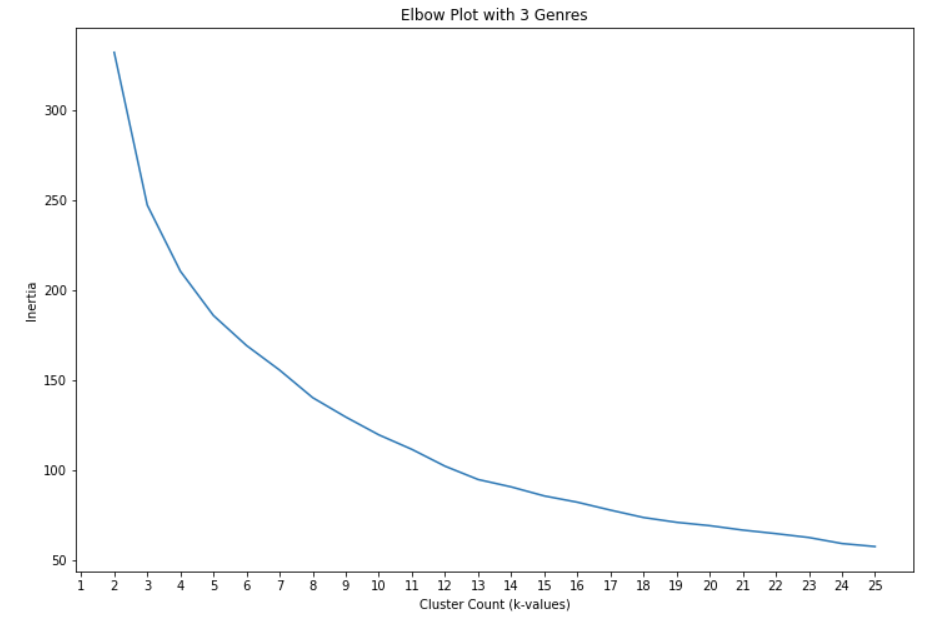


FIGURE 5

Figure 5 shows the elbow plot for a k-means clustering model using 3 genres (action, comedy, drama) and reduced dataset

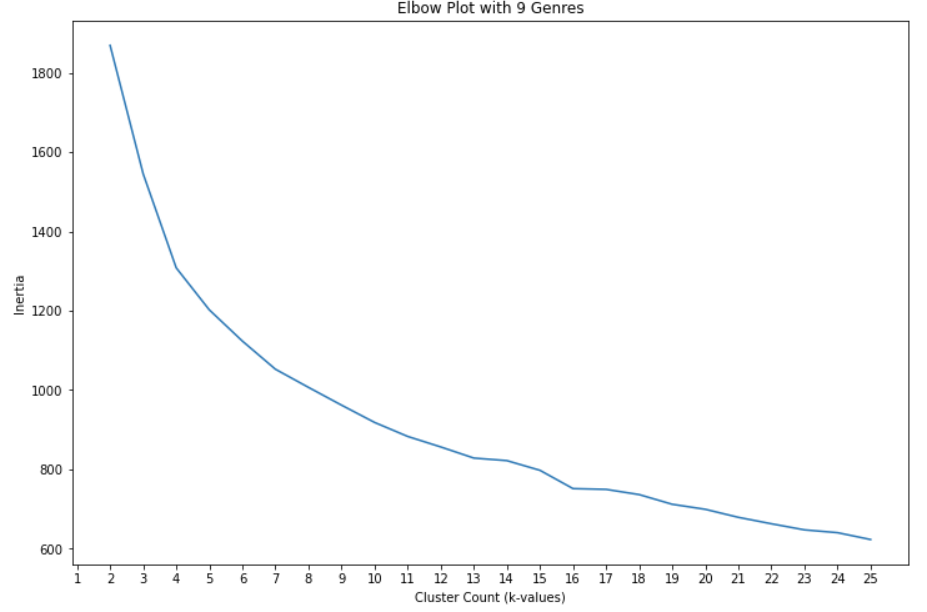


FIGURE 6

Figure 6 shows the elbow plot for a k-means clustering model using 9 genres ('Action', 'Adventure', 'Comedy', 'Crime', 'Drama', 'Mystery', 'Romance', 'Sci-Fi', 'Thriller') and reduced dataset

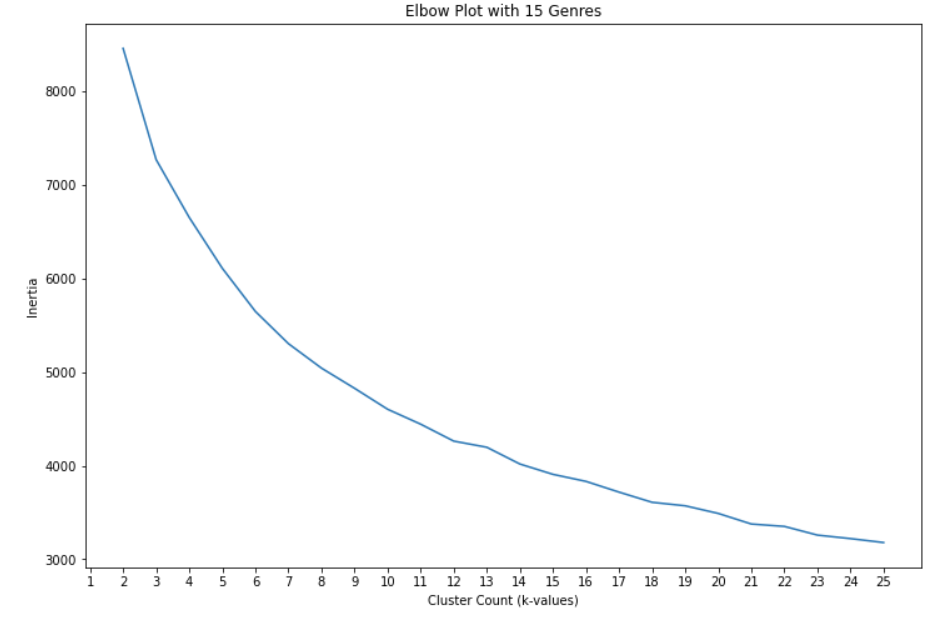


FIGURE 7

Figure 7 shows the elbow plot for a k-means clustering model using 15 genres ('Action', 'Adventure', 'Animation', 'Comedy', 'Crime', 'Drama', 'Fantasy', 'Horror', 'Musical', 'Mystery', 'Romance', 'Sci-Fi', 'Thriller', 'War', 'Western')

Conclusions: The elbow method (figures 5-7) is not a particularly accurate way to choose our k-value for k-means clustering as there is no definitive elbow or turning point in any of the graphs. The inertia does begin to level out as cluster number increases, so we focused on k-values around 10-12 which is within the range where the inertia still has a significant decrease between values and then considered the silhouette coefficient (figure 4). Surprisingly, the clusters using 9 genres are the least separate as k-values increase, while 3 genres give the most separation and 15 genres in the middle. We will focus our algorithm we apply to the full dataset on larger numbers of genres and around 12 clusters.

REFERENCES

*1] Carlos A. Gomez-Uribe and Neil Hunt. 2015. The Netflix recommender system: Algorithms, business value, and innovation. ACM Trans. Manage. Inf. Syst. 6, 4, Article 13 (December 2015), 19 pages. DOI: http://dx.doi.org/10.1145/2843948*

*[2] Eyrun A. Eyjolfsdottir, Gaurangi Tilak, Nan Li (2008), “MovieGEN: A Movie Recommendation System”, 2008 Conference Proceedings.*

*[3] Roman, Victor (2019), “Unsupervised Classification Project: Building a Movie Recommender with Clustering Analysis and K-Means”, Towards Data Science, https://towardsdatascience.com/unsupervised-classification-project-building-a-movie-recommender-with-clustering-analysis-and-4bab0738efe6*

*[4] Nixon, Alex Escola (2020), "Building a movie content based recommender using tf-idf", Towards Data Science, https://towardsdatascience.com/content-based-recommender-systems-28a1dbd858f5*