Logo

Description automatically generated**NILE UNIVERSITY**

**Movie Recommendation System**

A Math-301s Project Report

By

**Anas Ahmed - 202000005**

**Eslam Mohammed - 202000039**

**Mariam Barakat - 202000210**

**Yara Mahfouz - 202001787**

**Youssef Hisham - 202000543**

**Ziad Elkady - 202001598**

Submitted in partial fulfilment of the requirements

for Math-301 Project

To Dr. Marwa Sorour

**10 December 2022**

# ABSTRACT

A recommendation system is a system that provides recommendations to users for certain resources like games, movies, and music; this is based on data of the content or users. Such recommendation systems are beneficial for organizations such as streaming services that collect data from large amounts of customers to effectively provide the best suggestions possible, hence computer scientists are endeavoring to improve recommendation systems. Using the MovieLens dataset, we created two collaborative-based filtering systems that work with k-nearest neighbours and Singular Value Decomposition, and we compared and evaluated their effectiveness to find the most accurate approach. It is concluded that SVD is the most accurate algorithm.

Table of contents

[ABSTRACT 2](#_Toc123693586)

[I. Introduction 7](#_Toc123693587)

[II. Literature Review 8](#_Toc123693588)

[A. Content-based Filtering 8](#_Toc123693589)

[1) Content-based filtering algorithm 8](#_Toc123693590)

[2) Advantages and disadvantages of Content-based filtering 12](#_Toc123693591)

[B. Collaborative Filtering 12](#_Toc123693592)

[1) Memory Based 13](#_Toc123693593)

[2) Model-Based 16](#_Toc123693594)

[B. Hybrid-based Filtering 17](#_Toc123693595)

[1) Hybrid-based filtering algorithm 17](#_Toc123693596)

[2) Advantages and disadvantages of hybrid-based filtering 18](#_Toc123693597)

[C. Recommendation systems Applications 19](#_Toc123693598)

[1) Social Media recommendation system 19](#_Toc123693599)

[2) Movie recommendation system 19](#_Toc123693600)

[III. Methodology 20](#_Toc123693601)

[A. Dataset 20](#_Toc123693602)

[B. Approach 20](#_Toc123693603)

[C. KNN 21](#_Toc123693604)

[D. Evaluation 22](#_Toc123693605)

[1) Root-mean-square deviation 22](#_Toc123693606)

[2) Mean Absolute Error 22](#_Toc123693607)

[3) Empirical Analysis 22](#_Toc123693608)

[IV. Results AND Discussion 23](#_Toc123693609)

[A. K-Nearest Neighbors 23](#_Toc123693610)

[V. Conclusion 27](#_Toc123693611)

[References 28](#_Toc123693612)

[APPENDIX 31](#_Toc123693613)

[WORK DISTRIBUTION 32](#_Toc123693614)

**Table of Figures**

[Fig*.* 1*.* Content-based recommendation clarification diagram. 8](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693615)

[Fig. 2. One-hot encoding of movies and their genres. 9](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693616)

[Fig. 3. Pearson's correlation graphically represents the type of relationship according to the value of the correlation [10]. 10](#_Toc123693617)

[Fig. 4. Cosine similarity angles and the type of similarity it has. The closer the angle between the two vectors to zero, the more similar the vectors [10]. 10](#_Toc123693618)

[Fig. 5. Jaccard similarity represented in a Venn diagram [10]. 11](#_Toc123693619)

[Fig. 6. Manhattan Distance graphical representation compared to the Euclidean distance [6]. 12](#_Toc123693620)

[Fig. 7. Classification of collaborative filtering algorithms [14]. 13](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693621)

[Fig. 8. Difference between user-based and item-based filtering [15]. 14](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693622)

[Fig. 9. User-based filtering algorithm [16]. 15](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693623)

[Fig. 10. Content-based filtering algorithm [16]. 16](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693624)

[Fig*.* 11*.* Parallel design [25]. 17](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693625)

[Fig. 12. Sequential design [25]. 17](#_Toc123693626)

[Fig. 13. Weighted Hybridization [24]. 18](#_Toc123693627)

[Fig. 14 All KNN metrics used 23](#_Toc123693628)

[Fig. 15 Comparing original datapoints with predicted datapoints from KNN cosine 23](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693629)

[Fig. 16 Comparing original datapoints with predicted datapoints from KNN Jaccard 24](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693630)

[Fig. 17 Comparing the predicted datapoints using KNN Euclidean distance and the original datapoints 24](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693631)

[Fig. 18 Comparing original datapoints with the predicted datapoints using KNN Manhattan metric 25](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693632)

[Fig. 19 Barchart of KNN metrics and SVD evalution in RMSE and MAE 26](file:///C:\Users\Mariam%20Barakat\Documents\university\courses\linear%20algebra\Recommendation%20system\Final_paper_v4.docx#_Toc123693633)

**Table of Equations**

1. [9](#_Toc120906334)
2. [9](#_Toc120906335)
3. [10](#_Toc120906336)
4. [11](#_Toc120906337)
5. [11](#_Toc120906338)
6. [11](#_Toc120906339)
7. [14](#_Toc120906340)
8. [16](#_Toc120906341)
9. [18](#_Toc120906342)
10. [18](#_Toc120906343)
11. [18](#_Toc120906344)

**Table of Tables**

[TABLE 1 19](#_Toc123693634)

[Table 2 Evaluation of KNN metrics 25](#_Toc123693635)

# I. Introduction

With the rapid growth in digital information and the increasing number of internet visitors, a problem of information overload arises. Such a problem prevents users from easily accessing their items of interest on the internet. Therefore, a vital need for information filtration systems was observed. Despite information retrieval systems, like google, partially solving the problem, they still lack information personalization [1]. This has resulted in strong demand for recommender systems. Recommender systems are information filtration techniques that select information from dynamically generated data according to the user’s preferences or observed behavior about an item [1].

Recommender systems can be classified into two categories: personalized and non-personalized. Non-personalized systems are developed to provide recommendations to new users with sparse information on their personal preferences, such that the items are recommended based on their popularity and relevancy among other users of similar demographic [2]. On the other hand, personalized systems account for the user’s preferences and then recommend items that are tailored according to those preferences [2]. The user’s preferences are gauged by collecting implicit and explicit feedback to know whether the customer has liked an item or not [3]. To clarify, implicit feedback is recording the user’s behavior towards certain items, how many times he bought them, or the amount of time spent scrolling on related items [3]. However, explicit feedback is when the system records the user's rating and direct feedback, like an item survey or rating [3]. Both implicit and explicit feedback is used to create effective personalized systems. Furthermore, the flow of creating a personalized system is as follows: explicit and implicit feedback is collected about the user, and descriptive attributes are collected for the items. The latter data is then pre-processed in a way that enables the system to make predictions on what the user might also like [4]. The presence of sufficient user data, personalized recommender systems are always the better option. Boudet et al. [5] revealed that a successful implementation of personalization caused a 5% to 15% increase in revenue and a 10% to 30% marketing efficiency enhancement. So, given the major benefits that personalization adds to the user experience and the market gains, it can be concluded that more effort should be directed towards developing more efficient and more accurate personalized recommender systems.

This paper will focus on personalized recommendation systems and their applications. The literature review will first cover the different types of personalized recommender systems, namely content-based filtering, collaborative filtering, and hybrid filtering. It will discuss the steps taken to perform each algorithm, and their advantages and disadvantages. The differences between each type will also be highlighted. Moreover, two recommender systems applications, one for movies and the other for social media, will be explored. This research aims to implement a movie recommender system using collaborative filtering methods, one of which is implemented using singular value decomposition, and the other is KNN implemented with various similarity metrics. Both implementations will be evaluated and compared in terms of accuracy and efficiency, to determine which system is more suitable for solving the problem at hand.

# II. Literature Review

## A. Content-based Filtering

Content-based filtering is a way to recommend items according to the customer’s previous actions and feedback, in which items with high similarity with liked items will be recommended [4]. It is noted that clear data about the item’s description and user preference, like rating, clicks and likes, is needed [6]. Fig.1 clarifies the general concept of content-based filtering, in which the user gets recommended items that are like the items they have liked in the past.

Similar products

Recommended to user

Fig*.* 1*.* Content-based recommendation clarification diagram.

### Content-based filtering algorithm

It starts with tables that contain the item’s descriptions and the user’s preferences. To be able to compare items and find the most similar items, we need to convert the data into numerical representation. Subsequently, we then compare items preferred by a user with all items not yet seen using a mathematical formula to define similarity. Lastly, we order the items according to the similarity, the higher the similarity, the more likely the user will like this item.

#### Vectorization:

Vectorization is the process of transforming words into vectors [7]. This is an important step as we will apply the similarity equations on those vectors to be able to extract similar items. However, not all words are vectorized the same.

The first technique that will be discussed is One-hot encoding. In this technique, multiple vectors equal to the size of the vocabulary variations are created, and if this word is present, we increment the count in this dimension, Fig.2 clarifies the transformation. This results in a huge number of vectors that are then compared to find similarities [8]. Therefore, the dimensionality increases as the cardinality increases, meaning it is memory expensive.

Table

Description automatically generated

Fig. 2. One-hot encoding of movies and their genres.

TF – IDF, short for Term Frequency Inverse Document Frequency, is an algorithm that vectorizes documents in a way to value words that appear many times in the document and are relevant to the document and undervalue other words [9]. Therefore, TF-IDF is more efficient as it does not only focus on the repetitions of a word but also its importance. Moreover, TF-IDF will have lower dimensions as it will disregard irrelevant and low-in-count words.

(1)

Eq. (9) [9] shows the Term Frequency which is the ratio of repetition of a certain word to the total number of words in the document . Then divide by the log of the ratio of documents that have this word to the total number of documents . The log is used to normalize the denominator, meaning that TF-IDF has a range of 0 to 1.

#### Similarity Metrics

: After vectorizing our attributes, there is a need to compare the similarity between the items. There are myriads of ways to calculate the similarity, below are the most used methods.

* Similarity-based metrics

Pearson’s correlation is a mathematical formula to find the relationship between two continuous variables, the result is always between -1 and 1 inclusively [10]. Fig. 3 [10] illustrates that when the correlation is -1, the relationship is inversely proportional. However, when it is 1 it is exactly correlated. Moreover, the closer to 1 the more similar, the closer to -1 the more different, and the closer to 0 the more there is no relationship.

(2)

In eq. (2) [10], the n represents the number of data points, the or represents the or respectively from the dataset. Also, or is the mean of the x-points or y-points respectively.

Chart, line chart

Description automatically generated

Fig. 3. Pearson's correlation graphically represents the type of relationship according to the value of the correlation [10].

Cosine similarity is a way where we find the cosine of angles between two real-valued vectors, where the result is between 0 and 1 [10]. Fig. 4 [10] describes that 1 means similar, thus the more the vectors are similar the closer the cosine of the angle to 1. Furthermore, when cosine the angle is 0, meaning the vectors are orthogonal this means the vectors have no similarity.

(3)

Eq. (3) [10] represents the calculation of the cosine similarity. The and are the vectors where we find the dot product and divide it by the product of the norm of each vector.

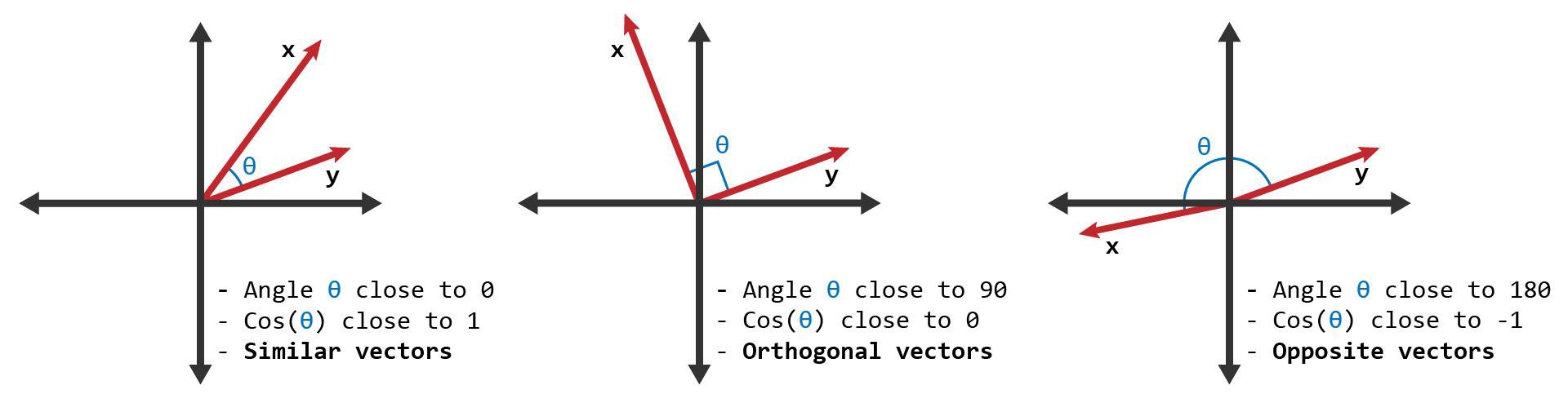


Fig. 4. Cosine similarity angles and the type of similarity it has. The closer the angle between the two vectors to zero, the more similar the vectors [10].

Jaccard similarity is used to calculate the similarity of two binary vectors [10]. It can be used for text similarity as the cosine similarity; however, it is more computationally expensive than cosine similarity as it matches all terms in one document to the other. Jaccard equation is represented in eq. (4) [10] , where A intersection B is shown in Fig.5 [10] in purple at the left, where A union B is the purple Venn diagram at the right.

Diagram, venn diagram

Description automatically generated

Fig. 5. Jaccard similarity represented in a Venn diagram [10].

(4)

* Distance Based Metrics

Euclidean distance is to find the straight line between two vectors, the less the distance the more similar. This method is not appropriate for text similarities, it is more relevant to continuous numerical variables. Also, it is preferred to normalize the vectors before finding the distance to be able to compare [10].

(5)

Eq.(5) [10] illustrates how to find the distance between two vectors, this will be by finding the squared difference of each and component in all the points available. The represents the number of data points, the and are the vectors and and are the points in the and components respectively.

Manhattan Distance is another method to get the distance, however, this is the distance from one vector to another [10]. The sum of the absolute difference between the measures in all dimensions of two points, as represented in eq. (6) [10]. Fig. 6 [10] illustrates the difference between the Manhattan distance and the Euclidean distance equation, where Manhattan is not the direct route to a vector.

(6)

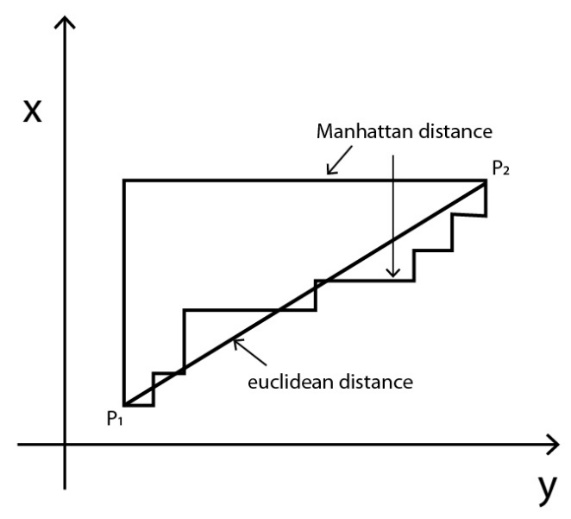


Fig. 6. Manhattan Distance graphical representation compared to the Euclidean distance [6].

#### Ordering

: Finally, after having a matrix with similarities, order the matrix in ascending order. The higher the similarity, the more likely the item will be liked by the user.

### Advantages and disadvantages of Content-based filtering

It is fortunate that in a content-based system, the model does not need any data about other users, which makes it easier to scale without the need for many users’ information. For this reason, it will capture specific interests and can recommend niche items that are not common to other people [11]. Moreover, new items can also be suggested without waiting for other people’s ratings as we depend on the description similarity to the user’s taste [6].

However, the model is not intelligent enough to expand on the user’s existing interests [11]. Furthermore, it requires a lot of domain knowledge, the model will be as good as the knowledge built on it [6]. Lastly, it is not effective when we do not have adequate information about new users to start making suggestions [6].

## B. Collaborative Filtering

As mentioned above, a content-based approach calls for a substantial amount of knowledge about the item’s description and user preference. Natural Language Processing, for instance, can be used to extract text from movie attributes like genre, year, director, actor, etc. So, in general, recommendations are treated as a user-specific classification problem and learn a classifier for the user's preferences based on the item's attributes.

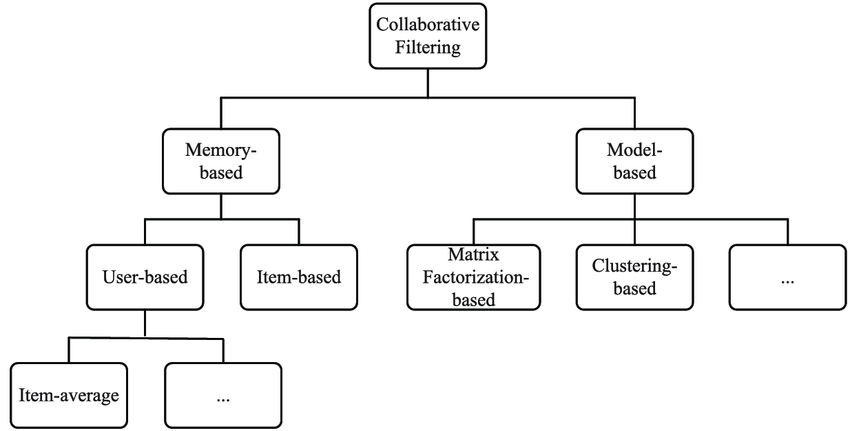
Collaborative Filtering (CF), on the other hand, is built by collecting users’ historical preferences on items and creating embeddings for each user and item [12]. It uses the combination of the user’s behaviour and compares that with other users’ behaviour in the database to make recommendations to users who share the same taste in items. Because it involves user interaction, collaborative filtering is the most known and used technique [13]. Moreover, several typical CF algorithms are classified as memory-based approaches and model-based approaches as illustrated in Fig. 7 [14].

Fig. 7. Classification of collaborative filtering algorithms.

Fig. 7. Classification of collaborative filtering algorithms [14].

### Memory Based

Only the user-item interaction matrix is used in memory-based filtering to provide users with updated recommendations. The overall procedure is based on the ratings and interactions the users have had previously. User-based collaborative filtering and item-based collaborative filtering are the two techniques used in memory-based filtering as shown in Fig. 8 [15], the picture portrays the different approaches that user-based and item-based collaborative filtering takes. As illustrated on the left part of the figure, user based looks for relationships between users. For instance, if two users have items in common, but one of them has an item that the other user does not have, this item will be recommended to him. On the other hand, and as illustrated on the right part of the figure, item-based looks for relationships between items which are represented as half-dotted lines. So, for example, if most users like a particular item also like another item and ones that likes one of the two items will get a recommendation for the other item.

#### a) User-Based

Fig. 8. Difference between user-based and item-based filtering [15].

In user-based filtering to give recommendations to a user, the nearest neighbours of that particular user are created based on his interactions, i.e., item ratings by the user. Items that are most popular to the neighbours of the user but new to the target user are used to be suggested for the target user [16]. As shown in Fig. 9 [16], if a recommendation needs to be made for a given user. First, both the target user and every other user are represented as vectors of interactions with different items. Then, the similarity between the target user and other users is such that each pair of users with the same interaction on items will be closer to each other [17]. A certain implementation of it is the user-based nearest neighbor algorithm. This algorithm needs two tasks:

1. Finding the K-nearest neighbours to the target user, using similarity function , such as cosine similarity as mentioned previously in eq. (3) [10] , to calculate the distance between each pair of users.
2. Assume that user a will rate every item that of their neighbours has consumed but has not. The -item with the highest predicted rating is the one we seek as illustrated in eq. 7 [15].

(7)

Graphical user interface, application

Description automatically generatedAlthough this technique is easy to implement and more accurate compared to content-based, the calculation of data for each pair of users required by a user-based strategy takes time. This algorithm is difficult to implement on large base platforms as a result [13]. Moreover, data sparsity is regarded as a significant drawback of user-based collaborative filtering. It is frequently believed that data scarcity may result in a lack of or a small number of co-rated items between two users, producing unreliable or unavailable similarity information and further resulting in poor recommendation quality [18].

Fig. 9. User-based filtering algorithm [16].

#### Item-Based

As discussed, user-based filtering suffers from several problems. To deal with these problems, we can use item-based collaborative filtering. Item-based filtering focuses on the items in the matrix that are more similar to what the target user positively rated and likewise, the similarity is computed similarity metric to predict the ratings of those items as illustrated in Fig. 10 [16]. In item-based two items are considered similar if both, while being represented as vectors are close to each other and assuming that a given user needs recommendations, the item vector that the user likes is represented by interactions with every user, then the similarity between this item and every other item is computed to get suggestions [16], [17]. In comparison to user-based filtering, this method uses fewer resources. The algorithm takes a lot less time with a new user than the user-based approach because similar values from all users are not acquired.

### Chart, diagram Description automatically generatedModel-Based

Fig. 10. Content-based filtering algorithm [16].

The model-based technique uses machine learning models to evaluate and rank user interactions with items with which they have not yet associated. These models are developed via a variety of algorithms, such as matrix factorization, deep learning, clustering, etc., that make use of the interaction data that is already there in the interaction matrix [16]. Moreover, a model-based can be used for both recommending items and predicting ratings for unrated items. This approach helps reduce the memory overhead that the memory-based approach has and is very useful when dealing with sparse matrices [18]. The clustering method attracts to restrict the data from expanding, they use repeated clustering in conjunction with the K-means clustering approach and the Expectation Maximization algorithm. In addition, Using the item descriptions provided on the web, stacked denoising autoencoders architecture may train the features for the deep learning technique, which can then utilize these features to be incorporated into the model. Neural collaborative filtering uses a multi-layer perceptron to learn the interactions between users and items, as it can learn any continuous function and has a prominent level of nonlinearities [19] which makes it affluent to learn from interactions.

SVD divides the initial matrix into submatrices containing latent features, which are not actual features present in the dataset but rather what the algorithm magically discovered as valuable hidden features [20]. It predicts the unknown ratings in a user-item matrix as shown in eq. (7) [21] using the inner product of **U** (user-user matrix), **Σ** (user-item matrix) and **VT** (item-item matrix transposed). Although only the k-number of features can be used to approximate the original matrix **M**, full singular value decomposition considers all m x n features in computation. This is the concept behind model-based collaborative filtering's dimensionality reduction. In addition, assuming that the feature matrix is folded into **U** and **VT** when used to recommender systems, we may anticipate the "approximated" rating matrix **M'** by the inner product of **U** and **VT**.

(8)

The product of **U** and **VT** is then used in a weighted way based on feature importance to fill in the missing ratings by minimizing the error, these latent feature parameters are trained iteratively [20].

## Hybrid-based Filtering

Hybrid-based filtering is a technique that combines multiple filtering techniques to recommend items. In collaborative filtering, there is a lack of knowledge regarding domain dependencies, as well as about people's preferences in a content-based system, a combination of both leads to an increase in common knowledge, when comparing hybrid recommender systems to collaborative or content-based systems, hybrid systems typically have superior recommendation accuracy [22]. The concept behind hybrid approaches is that a combination of algorithms will produce more accurate and effective suggestions than a single algorithm since the shortcomings of one algorithm can be mitigated by another [1].

Hybrid-based filtering techniques fall under two designs: parallel and sequential [23]. The parallel design feeds numerous recommendation systems, and each of those recommendations is integrated to produce a single output as in Fig. 11 [23]. The sequential design sends the input parameters to a single recommendation engine as in Fig. 12 [23].

Diagram

Description automatically generated

Fig*.* 11*.* Parallel design [25].

Diagram

Description automatically generated

Fig. 12. Sequential design [25].

### Hybrid-based filtering algorithm

One of the parallel design techniques is weighted hybridization. In this technique, weighted hybridization combines the outcomes of multiple recommenders to create a suggestion list or prediction by applying a linear equation to integrate the scores from each technique in use [1]. The weighted recommendation system will take the results of each model and aggregate them into static weightings as seen in Fig.13 [24], the weight of which will not change between the train and test sets [24].

Diagram

Description automatically generated

Fig. 13. Weighted Hybridization [24].

The weighted hybrid technique computes the prediction score as the results of all recommendation approaches by considering them as variables in a linear combination. This technique gives each of them weights and sums up the weighted results.

Weighted hybrid technique works by inputting the user profile in each recommendation system to generate an individual prediction as a score, then it combines the scores of each recommendation approach by considering them as variables in a linear combination [25]. Suppose that we want to combine collaborative and content-based filtering approaches to be combined using a hybrid weighted technique, the prediction score of users to item can be computed as follows:

(9)

Eq. (9) [9] shows the linear combination where denotes the weight of the algorithm . The symbol c denotes the number of used approaches, since we are interested in combining two recommendation approaches (collaborative and content-based approach), we set . The computation of the prediction score can be written as follows since as eq. (10) [25]:

(10)

The optimized weight can be computed as eq. (11) [25]

(11)

### Advantages and disadvantages of hybrid-based filtering

Hybrid systems combine different models to compensate for the shortcomings of one model thus overcoming the shortcomings of each algorithm and enhancing system performance [23]. Techniques such as clustering, similarity, and classification can be also utilized to generate more accurate and precise recommendations [22].

However, the hybrid model typically has high computational complexity and necessitates a huge database of ratings and other data to stay consistent [23].

## Recommendation systems Applications

Recommendation systems are intended to provide users with an individualizedonline product or service recommendations. More and more online businesses are using recommendation systems to increase user interaction and the likelihood of purchasing [26]. The use cases for recommendation systems have expanded rapidly over the past 4-5 years into various aspects of e-commerce, online media, and education, as described in Tab.1 [27].

TABLE 1

Companies that use recommendation systems [27].

Table

Description automatically generated

### Social Media recommendation system

As social media sites grow and their material quantities increase, users need help deciding which sites to participate in actively [26]. Social media and personalized recommender systems can commonly take advantage of one another; on one hand, social media presents new sorts of public information and metadata, such as tags, appraisals, comments, and explicit individual's connections, which can be utilized to improve recommendations [26]. Several personalized suggestion systems for social media have arisen in recent years. StumbleUpon1, for example, is a personalized recommender engine that proposes web pages based on a user's prior ratings or ratings from users with similar interests, and themes of interest chosen by the user from a list of almost five hundred categories [26]. Several social media platforms have recently included personalized suggestion features: for instance, a site for uploading videos like YouTube has introduced a customized homepage with suggestions based on previous views and favourites. This feature is said to have increased the number of users who visit the homepage [26], the frequency with which they visit, and the number of subscriptions they make over time.

### *Movie recommendation system*

 Recently, there has been widespread interest in recommendation systems that assist the selection process from various alternative proposals [27]. Recommender is a fusion of information retrieval and intelligence system research. Traditional movie sites work by displaying worldwide user ratings on films in their database, and the Metadata of movies, such as category, date, and producers. People may browse lists, search for movies, and read reviews by reviewers or other users. However, most of these services need a personal recommendation mechanism and have yet to tap into social networking networks or crowd knowledge. Developers must create systems with improved performance features and efficiency to match the similarities in client desires to seal product sales or movie watching [28].

In conclusion, there are 3 types of movie recommendation systems each with its advantages and disadvantages: Content-Based Filtering based, Collaborative Filtering, and Hybrid Filtering. A content-based recommendation system emphasizes the content features whereas a collaborative recommendation system emphasizes the user preference. In collaborative filtering, the user’s profile is required to suggest relevant content, unlike content-based filtering where a recommendation system uses the content profile too which includes the content features. The collaborative recommendation system feeds on the user and product data. So, products with no ratings or feedback cannot be recommended to any user. Neither a new user who has not given any reviews or ratings can get any recommendation by the collaborative recommendation engine. This is called the cold start problem. The content-based recommendation systems are product features oriented and hence do not have such problems. A content-based recommendation engine can provide more accurate recommendations as it focuses on the features of the content a user likes, whereas a collaborative recommendation engine does not always ensure precise recommendations because users with similar tastes may not always like the same products. Hybrid systems improve the results of the mode and increase the system's accuracy by overcoming the drawbacks of both content-based and collaborative filtering techniques. However, the hybrid model is much harder to perfect as it has high computational complexity and requires a huge database of ratings and other data to stay consistent.

# III. Methodology

## Dataset

The movie recommendation system will be implemented on a partition of the MoviesLens Dataset containing the metadata of 45,000 movies and the ratings of 270,000 users. The movie dataset contains movies released on or before July 2017 and has cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, counties, TMDB vote counts, and vote averages. In addition, the users' dataset contains the user’s ratings, and it is on a scale of 1 - 5. For now, the collaborative filtering will mainly account the users’ tables.

Table 2

Column names in user dataset, which will be used in collaborative filtering algorithms

|  |  |  |
| --- | --- | --- |
| UserID | MovieID | Rating |

## Approach

In the recommendation system user-based approach will be implemented. The proposed approach uses KNN and SVD algorithms to conduct rating prediction experiments. The paper will include the implementation of cosine similarity, Euclidean distance, and Manhattan distance. Moreover, we will use either the similarity or distance matrix to implement KNN on it. Lastly, an exploration of matrix factorization using SVD will be used before the implementation of KNN. All the proposed approaches will be compared and evaluated to conclude the best approach.

By using python libraries, after reviewing in details the movie dataset a model hs been made. The usage of NumPy, pandas, SciPy libraries and ScikitLearn in this proposed solution was for powerful performance on Jupyter Notebook python.

The following steps have been followed to carry out our implementation in both user-based and item-based methods:

1. Importing the data in the notebook.
2. Analyzing data to gain insights.
3. Splitting the ratings dataset to be 80% in the training dataset and 20% of the ratings in the test dataset.
4. Using the proposed algorithm to predict movies ratings and generate a score representing the prediction.
5. Compare the actual ratings with the predicted ratings of the user in the test dataset to give a proper evaluation of the algorithms.
6. Similar process is repeated with other algorithms for comparison.

## KNN

The K-Nearest Neighbors was implemented in the following steps:

1. Create a matrix
   1. Pivot the data by having the row named by users and the column names named by movies, then fill with the corresponding rating.
   2. Normalize the dataframe by finding the average rating, then subtract each rating by this average. Also, fill the Null cells with zeroes.
2. Create a mask on unnecessary parts of the matrix
   1. Locate the target user and movie.
   2. Find the users who watched the chosen movie.
   3. Filter out the users who watched the chosen movie.
   4. Mask the remaining section of the similarity matrix.
3. Find the datapoints to test on
   1. Find 50 datapoints which are not empty and store their positions in a list. This is done because the matrix is extremely sparse.
4. Implement the K-Nearest Neighbors.
   1. KNN with cosine similarity as a metric.
   2. KNN with Jaccard similarity as a metric.
   3. KNN with Manhattan distance as a metric.
   4. KNN with Euclidean distance as a metric.
5. Evaluation
   1. Graphed all KNN metrics on a graph.
   2. Graphed each KNN metric against the original data.
   3. Evaluated the KNN metrics using Mean Absolute Error and Mean Square Error Deviation.

## SVD

The SVD algorithm was implemented in the following steps:

1. Importing and prepearing data
2. Centering data using the mean and filling empty data with zeros
3. Applying SVD using scipy
4. Generating U, Sigma and
5. Getting the result of the dot product of U, Sigma and
6. Reshape the result to a matrix and adding the values before ceneting back
7. Masking the parts of the data and splitting it to 20% and 80% for testing
8. Evaluating the model using RMSE and MAE

## Evaluation

To analyze the performances of the proposed algorithms the following metrics are used:

### Root-mean-square deviation

Root mean squared error is the square root of the average squared distance (difference between actual and predicted value). It Is used in regression problems to estimate the rate of error and determine if there are any large errors caused by model overestimating or underestimating the prediction [29]. Eq.(12) [29] is the Root Mean Square Error where the summation of elements is brought, and actual and predicted are subtracted and squared, then the number is divided by the number of datapoints. Lastly, the product is square rooted.

(12)

### Mean Absolute Error

Mean absolute error is a formula used to calculate the degree of error comparing the predicted and the actual values of the datapoints. Eq.(13) [30] clarifies that this equation finds the average of all absolute errors. Where the i is the ith predicted y, and is the actual value.

(13)

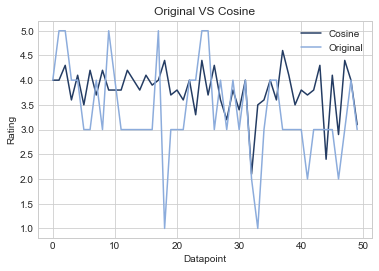
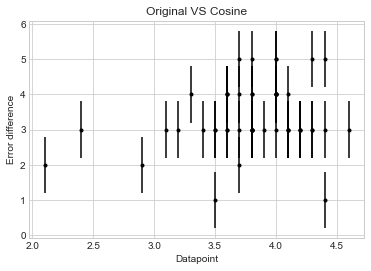
# IV. Results AND Discussion

Chart, line chart

Description automatically generatedFig. 14, illustrates an overall visual comparison between KNN metrics, namely, Cosine, Jaccard, Euclidean, Manhattan and the original data. It can be seen that all the prediction are fairly consistent with less variation compared to the original data. Also, it seems that Cosine similarity has the best prediction

Fig. 14 All KNN metrics used

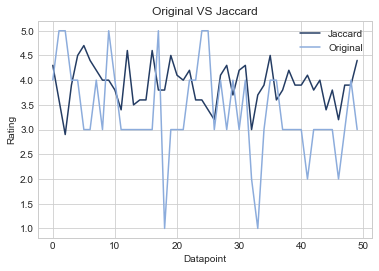
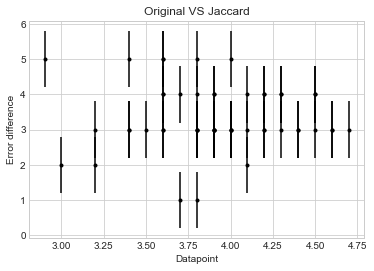
Fig. 15 displays the graphing of the original datapoints against the predicted datapoints using KNN, using cosine similarity as the metric. Fig. 15 (a), the line graph reveals that the cosine similarity metric predicts fairly well, as the line graphs look similar in shape; however, as shown in Fig. 15 (b) some points are predicted very poorly. This is because sometimes people may like movies that are not of their regular taste.



1. Original VS Cosine
2. Original VS Cosine Error comparison

Fig. 15 Comparing original datapoints with predicted datapoints from KNN cosine

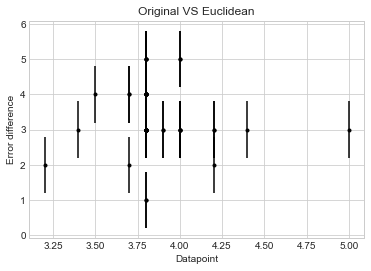
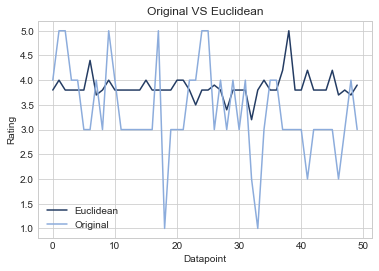
Fig. 16 desccribes the mapping of the prediction using KNN, specifically Jaccard metric, against the original datapoints. Fig. 16 (a) and (b) bluntly show how jaccard does a poor job at predicting. This maybe because to work with jaccard the datapoints must be converted to Boolean first, to be able to find its intersections and unions.



1. Original VS Jaccard
2. Original VS Jaccard Error comparison

Fig. 16 Comparing original datapoints with predicted datapoints from KNN Jaccard

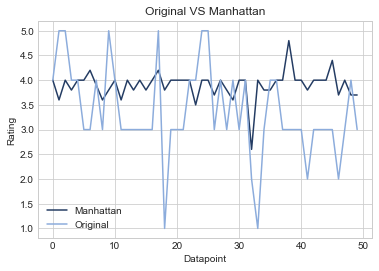
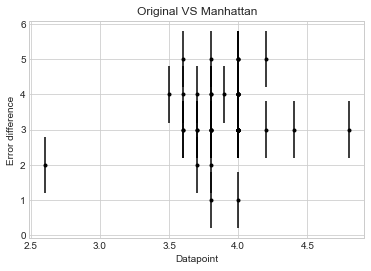
In Fig. 17 (a) – (b) it can be shown that the Euclidean metric predicts fairly well, but some points are very poorly predicted; it can be shown by the 2 downward spikes and the KNN Euclidean matric has predicted it poorly. It is noticed that the prediction has little variation, compared to the original data, which backs up the theory that human’s predictions are nonlinear.



1. Original VS Euclidean
2. Original VS Euclidean Error comparison

Fig. 17 Comparing the predicted datapoints using KNN Euclidean distance and the original datapoints

In Fig. 18(a), the graph of the Manhattan seems to have more variation compared to the Euclidean distance, the predictions are fairly well. However in Fig. 18(b) it reveals that the error focuses on one area.



1. Original VS Manhattan
2. Original VS Manhattan Error comparison

Fig. 18 Comparing original datapoints with the predicted datapoints using KNN Manhattan metric

Table 2 represents the Root Mean Squared deviation of Cosine, Jaccard, Euclidean and Manhattan metrics using the KNN impelementation in comparison with SVD. It seems that Cosine has the lowest RMSE between KNN implementations, meaning the best prediction among them. On the other hand SVD shows better results than all KNN implementations. Moreover, the table displays the Mean Absolute Error of the four metrics and SVD. The order of accuracy from best to worst is: SVD, Cosine, Manhattan, Euclidean, and finally jaccard with the poorest accuracy.

Table 2 Evaluation of KNN metrics and SVD

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Different Metrics | | | |
| SVD | Cosine | Jaccard | Euclidean | Manhattan |
| RMSE | 0.8636 | 1.0358 | 1.320 | 1.1532 | 1.1372 |
| MAE | 0.6526 | 0.7940 | 0.956 | 0.8920 | 0.8640 |

Fig. 19 visualises Table 2 using barchart with addition to SVD. It displays the superiority of Cosine similarity KNN metric on all other KNN metrics, however the SVD has superior accuracy compared to all. The SVD RMSE is better than the Cosine by 20%, better than the Jaccard by 53%, better than the Euclidean by 34%, better than the Manhattan by 32%. The cosine MAE is better than the Cosine by 22%, better than the Jaccard by 46%, better than the Euclidean by 37%, better than the Manhattan by 33%.

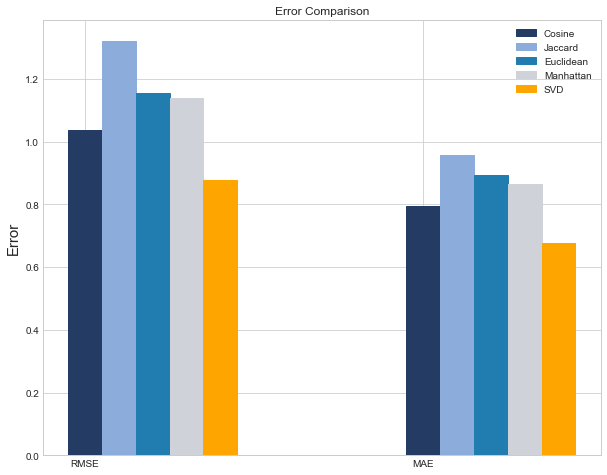


Fig. 19 Barchart of KNN metrics and SVD evalution in RMSE and MAE

# V. Conclusion

With the aim of finding the best technique for implementing a movies recommender system, two collaborative filtering models, namely KNN and SVD, were implemented, trained, and tested on the movielens dataset. Four variations of the KNN model were implemented using four different similarity metrics: cosine similarity, manhattan distance, euclidean distance, and jaccard similarity. After evaluating the outputs of all the models, a comparison between the results was conducted. It was shown that SVD had the lowest RMSE and MAE, with the second lowest RMSE and MAE belonging to KNN using cosine similarity. It was concluded that SVD is the better collaborative filtering technique for implementing a movies recommender system.

For future work, in order to enhance the prediction accuracy and lowering the error an ensemble learning approach will be used to combine models with the best performance. Moreover, a hybrid filtering technique which combines both content based and collaborative based filtering will be used.

# References

|  |  |
| --- | --- |
| [1] | F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, “Recommendation systems: Principles, methods and evaluation,” *Egypt. Inform. J.*, vol. 16, no. 3, pp. 261–273, 2015. |
| [2] | S. Khatwani and M. B. Chandak, “Building Personalized and Non Personalized recommendation systems,” in *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*, 2016, pp. 623–628. |
| [3] | A. Roy, “Introduction to recommender systems- 1: Content-based filtering and collaborative filtering,” *Towards Data Science*, 29-Jul-2020. [Online]. Available: <https://towardsdatascience.com/introduction-to-recommender-systems-1-971bd274f421> . [Accessed: 29-Nov-2022]. |
| [4] | B. Mobasher, “Data Mining for Web Personalization,” in *The Adaptive Web*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 90–135. |
| [5] | J. Boudet, B. Gregg, K. Rathje, E. Stein, and K. Vollhardt, “The future of personalization—and how to get ready for it,” *Mckinsey.com*, 18-Jun-2019. [Online]. Available: <https://www.mckinsey.com/capabilities/growth-marketing-and-sales/our-insights/the-future-of-personalization-and-how-to-get-ready-for-it> . [Accessed: 29-Nov-2022]. |
| [6] | “A guide to content-based filtering in recommender systems,” *Turing.com*, 05-Aug-2022. [Online]. Available: <https://www.turing.com/kb/content-based-filtering-in-recommender-systems> . [Accessed: 29-Nov-2022]. |
| [7] | “What is a vectorizer in nlp,” *ProjectPro*. [Online]. Available: <https://www.projectpro.io/recipes/what-is-vectorizer> . [Accessed: 29-Nov-2022]. |
| [8] | H. Heidenreich, “Natural Language Processing: Count Vectorization with scikit-learn,” *Towards Data Science*, 24-Aug-2018. [Online]. Available: <https://towardsdatascience.com/natural-language-processing-count-vectorization-with-scikit-learn-e7804269bb5e> . [Accessed: 29-Nov-2022]. |
| [9] | B. Góralewicz, “The TF\*IDF algorithm explained,” *Onely*, 06-Mar-2018. [Online]. Available: <https://www.onely.com/blog/what-is-tf-idf/> . [Accessed: 29-Nov-2022]. |
| [10] | M. Lüthe, “Calculate Similarity — the most relevant Metrics in a Nutshell,” *Towards Data Science*, 17-Nov-2019. [Online]. Available: <https://towardsdatascience.com/calculate-similarity-the-most-relevant-metrics-in-a-nutshell-9a43564f533e> . [Accessed: 29-Nov-2022]. |
| [11] | “Content-based filtering advantages & disadvantages,” *Google Developers*. [Online]. Available: <https://developers.google.com/machine-learning/recommendation/content-based/summary> . [Accessed: 29-Nov-2022]. |
| [12] | S. Luo, “Introduction to recommender system,” *Towards Data Science*, 10-Dec-2018. [Online]. Available: <https://towardsdatascience.com/intro-to-recommender-system-collaborative-filtering-64a238194a26> . [Accessed: 29-Nov-2022]. |
| [13] | S. Saha, S. Ramamoorthy, and E. Raghav, “User centric and collaborative movie recommendation system under customized platforms,” in *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*, 2021, pp. 699–704. |
| [14] | Z. Yang, B. Wu, K. Zheng, X. Wang, and L. Lei, “A survey of collaborative filtering-based recommender systems for mobile internet applications,” *IEEE Access*, vol. 4, pp. 3273–3287, 2016. |
| [15] | C. Pinela, “Recommender Systems — user-based and item-based Collaborative Filtering,” *Medium*, 06-Nov-2017. [Online]. Available: <https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f> . [Accessed: 29-Nov-2022]. |
| [16] | “How does collaborative filtering work in recommender systems?,” *Turing.com*, 29-Jul-2022. [Online]. Available: <https://www.turing.com/kb/collaborative-filtering-in-recommender-system> . [Accessed: 29-Nov-2022]. |
| [17] | H. Vijay, “Recommendation System using kNN,” *Auriga IT*, 10-Apr-2020. [Online]. Available: <https://www.aurigait.com/blog/recommendation-system-using-knn/> . [Accessed: 29-Nov-2022]. |
| [18] | B. Hu, Z. Li, and W. Chao, “Data sparsity: A key disadvantage of user-based collaborative filtering?,” in *Web Technologies and Applications*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 602–609. |
| [19] | A. Sharma, “Neural collaborative filtering,” *Towards Data Science*, 16-Dec-2019. [Online]. Available: <https://towardsdatascience.com/neural-collaborative-filtering-96cef1009401> . [Accessed: 29-Nov-2022]. |
| [20] | D. Gong, “An intro to collaborative filtering for movie recommendation,” *Towards Data Science*, 11-Dec-2021. [Online]. Available: <https://towardsdatascience.com/a-beginner-friendly-guide-to-recommender-system-3f5fa2a57c02> . [Accessed: 29-Nov-2022]. |
| [21] | “Singular Value Decomposition (SVD) tutorial,” *Mit.edu*. [Online]. Available: <https://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm> . [Accessed: 29-Nov-2022]. |
| [22] | G. Geetha, M. Safa, C. Fancy, and D. Saranya, “A Hybrid Approach using Collaborative filtering and Content based Filtering for Recommender System,” *J. Phys. Conf. Ser.*, vol. 1000, p. 012101, 2018. |
| [23] | Vatsal, “Recommendation systems explained,” *Towards Data Science*, 12-Jul-2021. [Online]. Available: <https://towardsdatascience.com/recommendation-systems-explained-a42fc60591ed> . [Accessed: 29-Nov-2022]. |
| [24] | J. Chiang, “7 types of hybrid recommendation system - analytics Vidhya - medium,” *Analytics Vidhya*, 26-Jun-2021. [Online]. Available: <https://medium.com/analytics-vidhya/7-types-of-hybrid-recommendation-system-3e4f78266ad8> . [Accessed: 29-Nov-2022]. |
| [25] | S. Suriati, M. Dwiastuti, and T. Tulus, “Weighted hybrid technique for recommender system,” *J. Phys. Conf. Ser.*, vol. 930, p. 012050, 2017. |
| [26] | I. Guy, N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel, “Social media recommendation based on people and tags,” in *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR ’10*, 2010. |
| [27] | M. Kumar, D. K. Yadav, A. Singh, and V. Kr., “A Movie Recommender System: MOVREC,” *Int. J. Comput. Appl.*, vol. 124, no. 3, pp. 7–11, 2015. |
| [28] | S. Jayalakshmi, N. Ganesh, R. Čep, and J. Senthil Murugan, “Movie recommender systems: Concepts, methods, challenges, and future directions,” *Sensors (Basel)*, vol. 22, no. 13, p. 4904, 2022. |
| [29] | J. Moody, “What does RMSE really mean?,” Towards Data Science, 05-Sep-2019. [Online]. Available: <https://towardsdatascience.com/what-does-rmse-really-mean-806b65f2e48e> . [Accessed: 08-Dec-2022]. |
| [30] | 20\_\_80\_\_, “Mean absolute error ~ MAE [Machine Learning(ML)],” Medium, 02-Feb-2018. [Online]. Available: <https://medium.com/@20__80__/mean-absolute-error-mae-machine-learning-ml-b9b4afc63077>. [Accessed: 08-Dec-2022]. |

# APPENDIX

KNN implementation [link](https://colab.research.google.com/drive/1OrBRLS-smqDwQAiFlrskoxx6zOjpsxlT?usp=sharing)

SVD implementation [link](https://colab.research.google.com/drive/1OXqbrS_r76ZXm-RcZOgMdMscNQwWAQZ8?usp=sharing)

# WORK DISTRIBUTION

|  |  |
| --- | --- |
| Anas Ahmed | Hybrid-based – poster- Presentation |
| Eslam Mohamed | Collaborative-based – SVD implementation |
| Mariam Barakat | Content-based – KNN implementation |
| Yara Mahfouz | Introduction – Conclusion and Results |
| Youssef Hisham | Movie recommendation applications - Presentation |
| Zaid Elkady | Conclusion – 2 page summary |