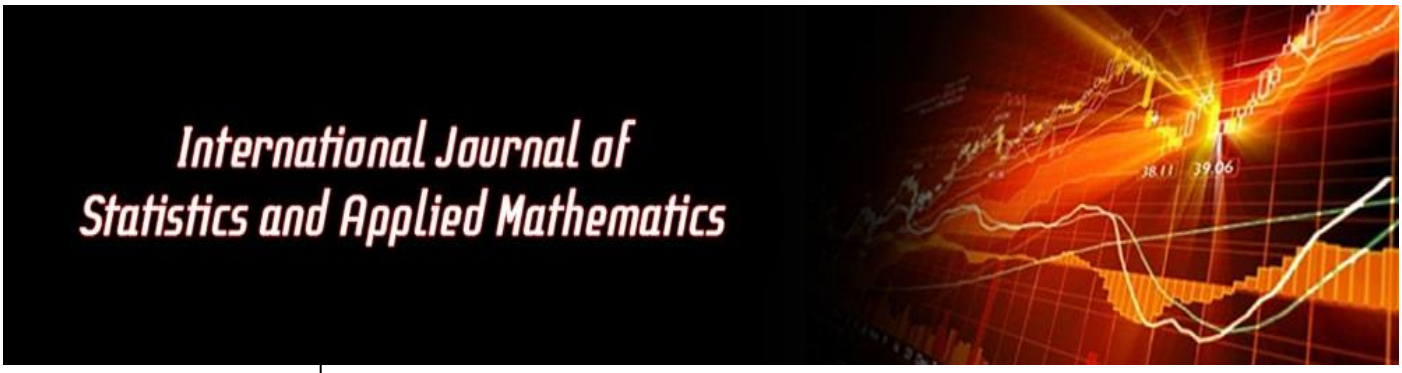
International Journal of Statistics and Applied Mathematics 2023; SP-8(3): 121-137



ISSN: 2456-1452
Maths 2023; SP-8(3): 121-137

https://www.mathsjournal.com
Received: 08-02-2023
Accepted: 09-03-2023

Amaan Mashooq Nasser
Vellore Institute of Technology, Vellore, Tamil Nadu, India

Jayant Bhagat
Vellore Institute of Technology, Vellore, Tamil Nadu, India

Abhishek Agrawal
Vellore Institute of Technology, Vellore, Tamil Nadu, India

T Joshva Devadas
Vellore Institute of Technology, Vellore, Tamil Nadu, India

Corresponding Author:
T Joshva Devadas
Vellore Institute of Technology, Vellore, Tamil Nadu, India

Mean-Reversion Based Hybrid Movie Recommender System Using Collaborative and Content-Based filtering methods

Amaan Mashooq Nasser, Jayant Bhagat, Abhishek Agrawal and T Joshva Devadas

DOI: https://doi.org/10.22271/maths.2023.v8.i3Sb.1012

Abstract
Machine Learning algorithms have a variety of important applications, and among them, Recommender systems are crucial. The internet hosts an extensive volume of information, making it challenging for users to navigate and find relevant content. Recommender systems have therefore emerged as valuable tools to bridge this gap. They facilitate the connection between users and relevant content by offering personalized recommendations. In recent years personalized recommendation service has become a hotspot of web technology, and is widely used in information, shopping, film and television, etc. [1]. Recommender systems have been proved to be an important response to the information overload problem [17].

In this research paper, we describe our approach for a Movie Recommender System Utilizing Mean Reversion via the Bollinger Bands formulae. Collaborative filtering is a popular technique used in Recommender systems. However, it poses a challenge in the form of the cold start issue, where new users are added to the system without any ratings, and the filter is unable to offer useful recommendations due to a lack of understanding of their preferences. Similarly, newly released movies without any ratings also suffer from the same issue, leading to recommendations reinforcing themselves.

To address this challenge, we incorporated the concept of Mean Reversion, which is a fundamental component of Natural Mathematics. Mean Reversion helps in mitigating the cold start issue by bringing new users and newly released movies into the fold of the Recommender system.

Mean reversion is a statistical concept that refers to the tendency of a series of values to return to its long-term average after experiencing temporary fluctuations. In the context of Recommender systems, Mean Reversion can be used to address the cold start issue by estimating the average rating for a movie and adjusting it based on a new user's preference. This technique can help improve the accuracy of recommendations, particularly for new users and newly released movies that lack sufficient data.

Keywords: Movie recommender system, mean reversion, content-based filtering, collaborative filtering

Introduction

A. Movie Recommender System

Movie recommender systems are a vital application of machine learning (ML) algorithms that predict or filter users' film preferences based on prior decisions and actions. In the current era, where the internet is an integral part of daily life, users frequently encounter the challenge of navigating an overwhelming amount of information. To assist users in this information boom, many organizations have implemented recommender systems. Despite decades of research on recommender systems, interest in these systems remains high due to their abundance of practical applications and the complexity of the subject. Movie recommender systems are especially important for corporate profits, with 66% of movies watched on Netflix in 2021 being recommended to users. Personalized recommendations of high quality enhance users' overall experience, providing new depth to their viewing options. For movie streaming services like Netflix, recommendation systems are important for helping users to discover new content to enjoy [6].

Recent years have seen the provision of personalized information to users through web-based recommendation systems of various types, with a wide range of applications. Recommender systems can be divided into two primary categories: those that use content-based filtering and those that use collaborative filtering. However, despite the success of recommender systems, a challenge remains regarding new users and new movies that lack ratings, commonly known as the "cold start" problem. Collaborative filtering is unable to provide useful recommendations because it is unaware of a user's interests. The same problem occurs with newly released films, which have yet to receive any ratings. As a result, recommendations may become self-reinforcing, leading to inadequate or irrelevant results.

This research paper focuses on a method of overcoming the "cold start" problem in a movie recommender system using mean reversion via the Bollinger Bands formulae. Mean reversion is a core concept of natural mathematics and is used to identify when an asset price is deviating from its average value, indicating that it is likely to return to its mean. By utilizing this approach, we aim to improve the accuracy and relevance of movie recommendations, enhancing users' overall experience.

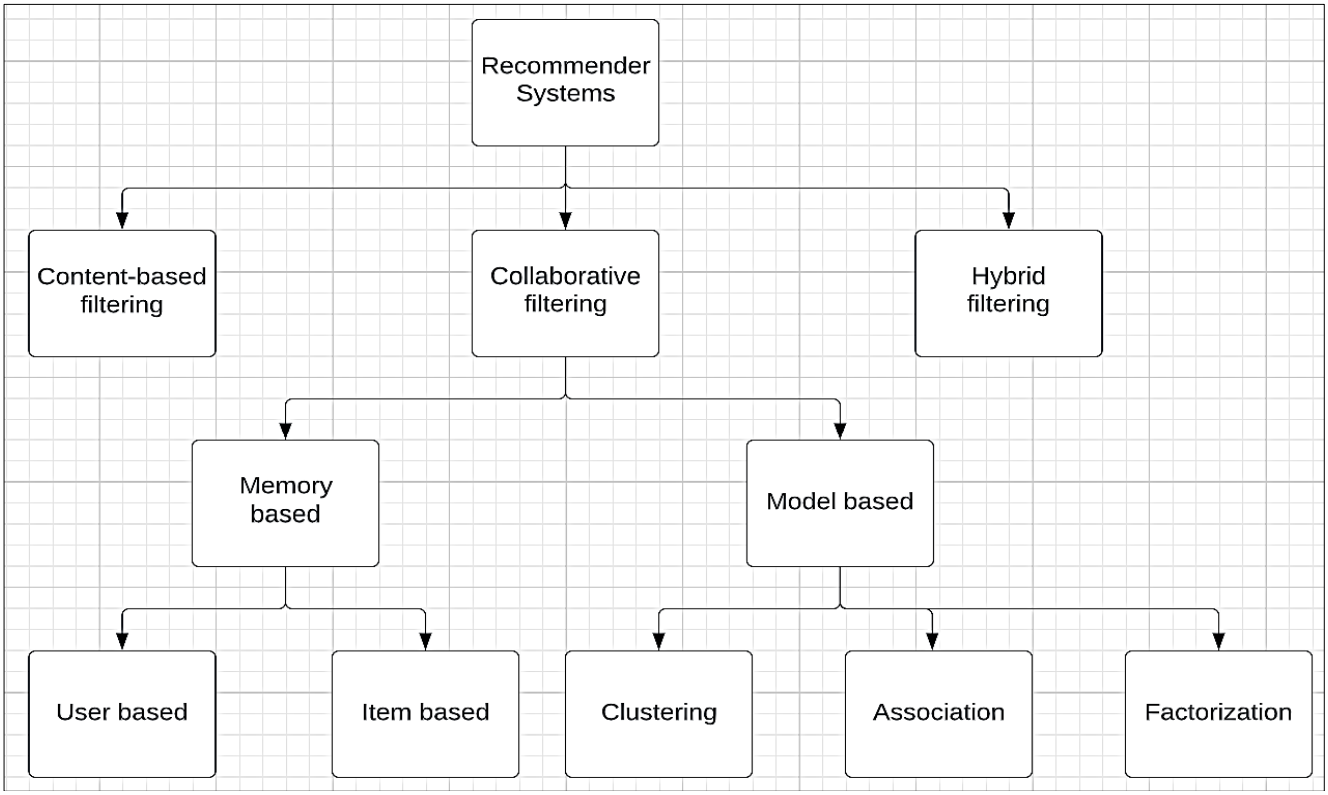


Fig 1: Taxonomy of Recommender Systems

B. Content-based filtering

Content-based filtering (CBF) relies on the user's preference profile and the item to make recommendations. CBF algorithms use keywords from the user's profile to represent an item's likes and dislikes. These algorithms prioritize items that the user previously liked or items that are related to those items. By examining the user's history of rated items, the CBF algorithm suggests the best-matched items. The user's profile is continuously updated as they add more information or interact more frequently with the recommended items, making the recommendation engine increasingly accurate over time.

A content-based recommendation system, which mainly relies on cognitive filtering, is “a system that displays item recommendations by comparing alternative items with those associated with user profiles. A prerequisite for content-based filtering is the availability of information about relevant content features of the items [14].

Balabanovic *et al*. had proposed a content-based recommendation system which can be applied in different domains, such as, books, movies, videos, or music. It uses different features, such as, author, genre, and most frequently used words [7].

Attributes shown in Fig 2 are explained as follows

- **User Profile:** The user's preferences are represented by vectors generated in the User Profile. To build the profile, we use a utility matrix that links the user and the item. By combining the attributes or tags of multiple user profiles, we can predict which movies are most likely to be preferred by the user.
- **Item Profile:** In content-based recommender systems, we create a profile for each item that represents its key elements. For movies, important characteristics include the performers, director, year of release, genre, and IMDb rating.
- **Utility Matrix:** The utility matrix indicates the user's preference for specific items. We use it to establish a connection between the user's preferred and least preferred items based on the data collected. Each user-item pair is assigned a "degree of preference" value in the utility matrix. By analyzing the matrix, we can determine the user's preferences for different items.

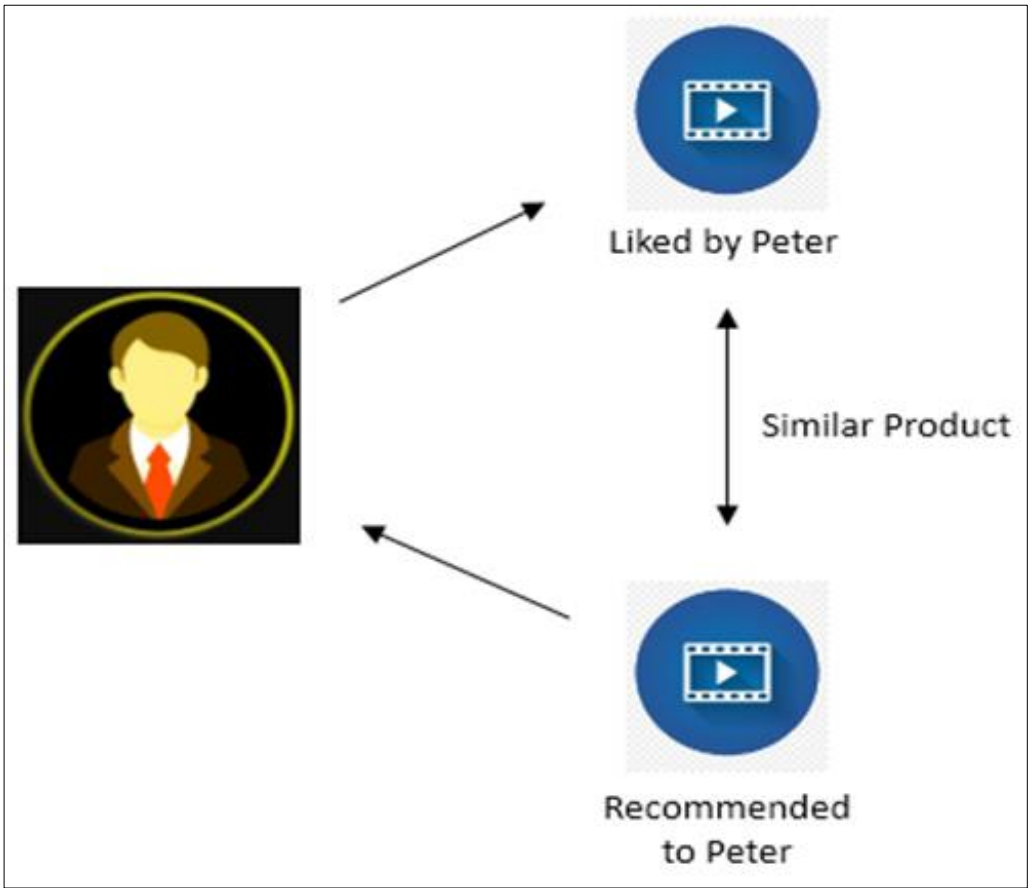


Fig 2: An illustration of content-based filtering

C. Collaborative filtering

Collaborative filtering (CF) is a widely used recommendation method in recommendation system since the mid-1990s [2]. This algorithm is based on the assumption that "Birds of a feather flock together", and users who like the same items are more likely to have the same interests [3]. It may also be used to find correlations among the objects rated [13].

Collaborative filtering is a recommendation approach that compares the similarities between users and items [11], addressing some of the limitations of content-based filtering. Collaborative filtering algorithms can suggest unexpected items to a user based on the preferences of other users who share similar interests. The embeddings used in collaborative filtering can be learned automatically, eliminating the need for manually designed features.

There are two categories of collaborative filtering: user-based and item-based. User-based CF measures the similarity between the target user and other users, while item-based CF measures the similarity between the items the target user interacts with and other items. The key idea behind collaborative filtering is that similar users have similar interests and preferences. It is getting more difficult to make a recommendation to a user about what he/she will prefer among those items automatically, not only because of the huge amount of data, but also because of the difficulty of automatically grasping the meanings of such data [10].

One of the challenges with collaborative filtering is the cold start problem, where new users or items without any ratings make it difficult for the system to provide meaningful recommendations. Self-reinforcing recommendations can occur for popular content with many ratings, while content with few ratings may not receive any recommendations. Cold start problem can be solved by content-based recommendation algorithm well since it does not need to provide historical behavior data of users. This algorithm mainly recommends the content to users according to the feature data of user and item provided in advance [8].

Collaborative neural graph filtering allows the system to improve its embedding process over time, enhancing its ability to make recommendations. We explored the use of LSTM-CNN recommender algorithms for movie recommendations by mining user behaviour data and recommending movies with higher ratings. We also studied hybrid models that use sentiment analysis and opinion mining to eliminate the need for users to browse through a large selection of films before making a selection. Memory-based approaches for collaborative filtering identify the similarity between two users by comparing their ratings on a set of items [12].

Overall, collaborative filtering is an effective approach for making recommendations based on user behaviour and preferences, and can be enhanced through the use of advanced techniques such as neural graph filtering and sentiment analysis.

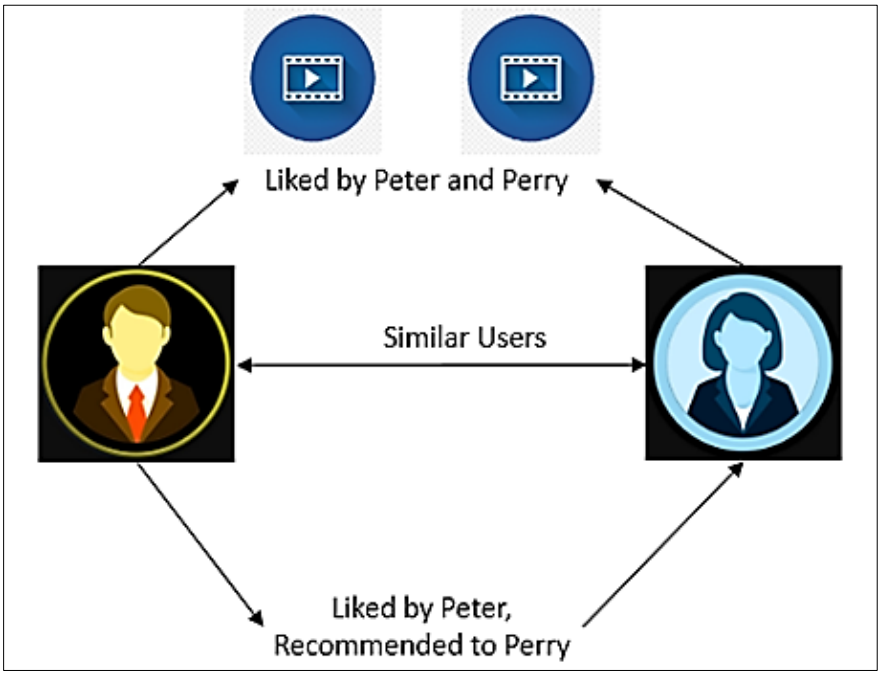


Fig 3: An illustration of Collaborative filtering

Table 1: Algorithms

Algo/Desc	Content-based
Content-based	Recommends based on user profile, interest, purchase history etc.
Collaborative	Recommends based on similarity of item or user
Hybrid	Hybrid method using Mean Reversion

III. Proposed methodology

To address aforementioned challenges, a hybrid model- based movie recommendation approach is needed to alleviate the issues of both high dimensionality and data sparsity [5].

This paper proposes a movie recommender system that combines content-based filtering and collaborative filtering algorithms. Collaborative filtering has traditionally been used to suggest movies based on user similarity in terms of genre and other movie tags. However, a limitation of collaborative filtering is the "cold start problem" where new users or movies without ratings cannot be recommended. Content-based filtering addresses this problem by suggesting movies based on their characteristics such as actors, directors, and genres. While existing algorithms combining both techniques have achieved respectable precision rates, this paper proposes an enhancement through the introduction of Natural Mathematics via Mean Reversion. This method allows the system to give more natural movie recommendations to the user.

In nature, most numerical values tend to revolve around a defined mean. Even when they do fluctuate, they do so in a fixed range that can be mathematically derived. This phenomenon is seen in many fields, from finance to geometry [18].

A Bollinger Band is a technical analysis tool defined by a set of trendlines. They are plotted as two standard deviations, both positively and negatively, away from a simple moving average (SMA) of a security's price and can be adjusted to user preferences [19].

They are curves drawn in and around the price structure usually consisting of a moving average (the middle band), an upper band, and a lower band that answer the question as to whether prices are high or low on a relative basis [21].

Accuracy of Bollinger Bands

Since Bollinger Bands are set to use +/- two standard deviations around an SMA, we should expect that approximately 95% of the time, the observed price action will fall within these bands [19].

Calculation of Bollinger Bands

First, calculate a simple moving average. Next, calculate the standard deviation over the same number of periods as the simple moving average. For the upper band, add the standard deviation to the moving average. For the lower band, subtract the standard deviation from the moving average [20].

Fig 4a: Ticker symbol of security

Fig 4b: Bollinger Bands drawn alongside the security

In figures 4a and 4b, we have presented an illustration of the security 'NIFTY AUTO' as of April 21st and its 1-minute chart. In addition, we have included the corresponding Bollinger Bands of NIFTY AUTO. We have used Trading View software to plot the charts and the Bollinger Bands.

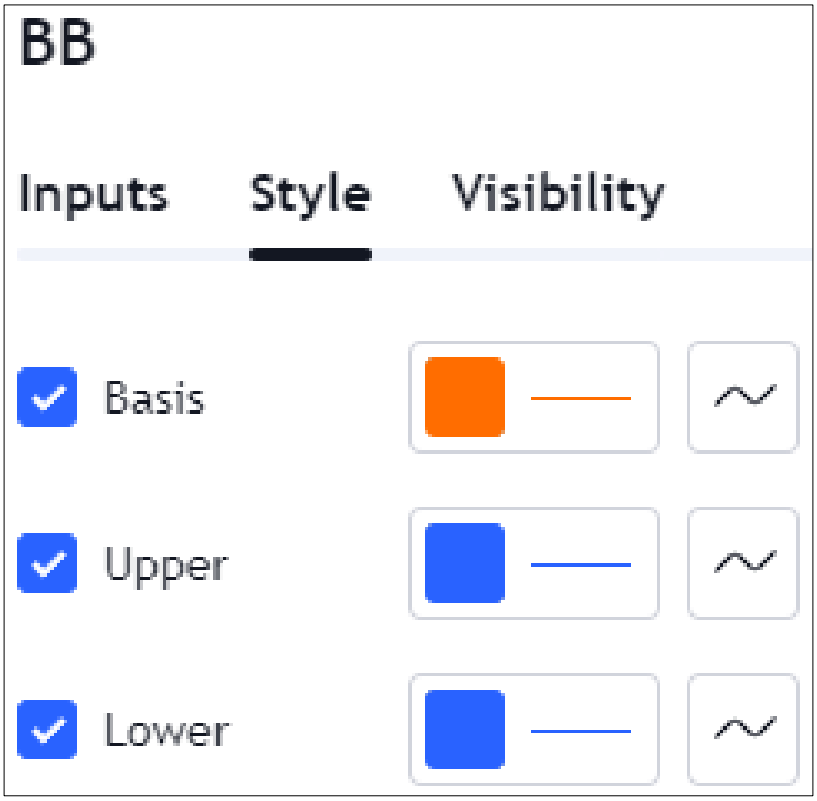


Fig 5a: Colour codes for the band and the average line

It is evident that the security's value is constrained within the predetermined range defined by the Bollinger Bands. The security tends to decrease in value when it touches the upper band, then moves towards the average line. On the other hand, the security tends to increase in value when it touches the lower band, then moves towards the average line.

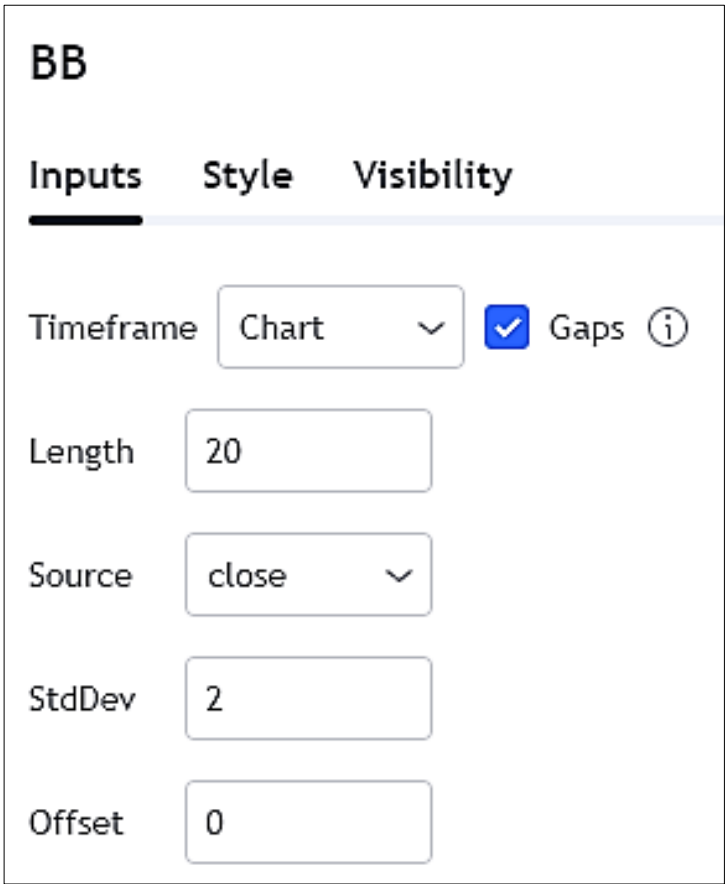


Fig 5b: Parameter of the Bollinger Band

In fig. 5b, we have shown the parameters used for the Bollinger Bands. The different parameters are:

a) **Timeframe:** This parameter specifies the timeframe to be used for plotting the Bollinger Bands, which is 1 minute in our case.
b) **Length:** This parameter indicates the number of price candles that are considered when calculating the average line.
c) **Source:** Each candle comprises of four price points, including the open price, timeframe-specific high price, low price, and close price. This parameter determines the price point used for calculating the average line and the corresponding Bollinger Bands.
d) **Std Dev:** This parameter specifies the value by which the standard deviation is multiplied to determine the upper and lower bands of the Bollinger Bands.
e) **Offset:** Altering this value will adjust the position of the Bollinger Bands either forwards or backwards concerning the current market. The default value is 0.

This paper employs mean reversion technique in the collaborative filtering algorithm. The technique addresses the popularity bias issue in collaborative filtering where popular movies tend to dominate recommendations. By considering the popularity trend of a movie, mean reversion technique brings balance to the recommendations, ensuring that niche or less popular movies are also suggested to users. The proposed system is implemented using the IMDb Movies Dataset and evaluated through experimental research. The results demonstrate the effectiveness and efficiency of the proposed system in providing accurate and diverse movie recommendations.

A. Approach & Design

In the field of recommendation systems, the use of advanced statistical techniques has become increasingly popular. One such technique is mean reversion, which describes how an asset's price tends to move back towards its average or mean over time. This statistical phenomenon has been observed not only in finance but also in various fields, including physics, biology, and engineering. In finance, mean reversion has been shown to hold in the stock market, where stocks that have performed well in the past tend to have lower returns in the future, and stocks that have performed poorly in the past tend to have higher returns in the future.

The strategy of mean reversion has been employed by various quantitative trading firms and hedge funds to generate profits. This strategy involves identifying assets that have deviated significantly from their mean and taking positions that anticipate their return to the mean. The market's tendency to overreact to short-term events creates opportunities for profit by buying undervalued assets and selling overvalued ones. Mean reversion can also be applied in the context of recommendation systems, where it can be used to identify movies that have been overlooked or undervalued by users and recommend them to new users who are likely to enjoy them.

However, existing recommendation systems may not be utilizing all available information from user-item interactions. In particular, they may not be distinguishing between two types of attribute interactions: Cross interactions and inner interactions.

Cross interactions occur when user attributes and item attributes interact, while inner interactions only occur between user attributes or item attributes. To address this limitation, a neural Graph Matching based Collaborative Filtering model (GMCF) has been proposed. This model aggregates attribute interactions in a graph matching structure, explicitly carrying out characteristic learning and preference matching based on cross interactions and inner interactions, respectively.

In addition to the distinction between cross and inner interactions, content-based recommender systems rely on the features of the item that the user responds positively to in order to predict the features or behaviour of the user. The use of feature vectors created from genre, actor, and other attributes can be employed to embed the user in an embedding space and suggest movies based on what the user thinks about movies in that genre. Collaborative filtering algorithms, on the other hand, create embedding for users and movies in the same embedding area and take into account the opinions of other users when proposing a movie to a user. Item-item filtering techniques can also be employed to suggest comparable items to the user based on their prior preferences.

In conclusion, the incorporation of mean reversion and advanced statistical techniques, such as the neural Graph Matching based Collaborative Filtering model and content-based recommender systems, can improve the accuracy and effectiveness of recommendation systems. By taking into account the market's tendency to revalue certain items and distinguishing between different types of attribute interactions, these techniques provide a more personalized and tailored recommendation experience for users.

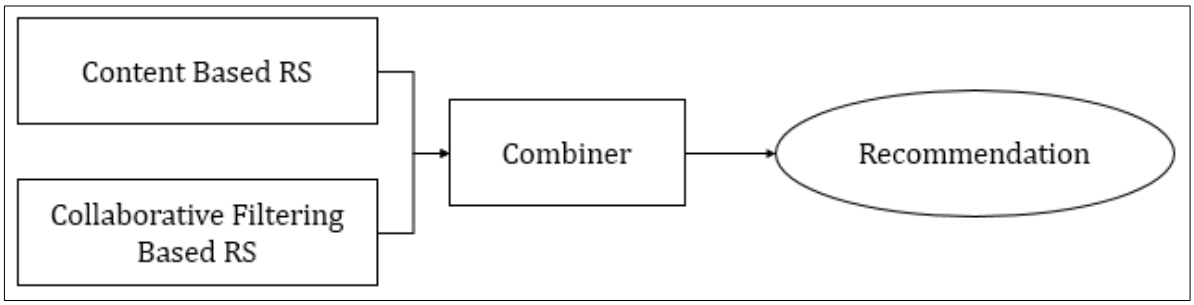


Fig 6: Working of recommender system

B. JAM Equation

We present JAM, a novel algorithm developed as a hybrid of two prominent filtering techniques, namely content-based and collaborative filtering. The term 'JAM' is an acronym derived from the first letters of the authors' names. The proposed algorithm represents a new formula that addresses the challenges posed by both approaches, aiming to improve the accuracy and efficiency of recommendation systems. Through a comprehensive evaluation, we demonstrate the superiority of JAM over existing state-of-the-art algorithms, highlighting its potential for practical applications.

The implemented system utilizes a unified module to compute the JAM score of matrices associated with content-based and collaborative filtering methods. In addition, two distinct modules are utilized to compute the scores for content-based and collaborative approaches. This design allows for a seamless integration of the two methods, resulting in more accurate and comprehensive recommendations for the end-users.

Using cosine similarity method calculate the scores for both the matrices. Using these scores, calculate the Standard deviation and Bollinger Band to produce hybrid score.

Let A = content-based matrix, B = collaborative matrix, [i, j] = mappings of each element in matrices A & B.

$$W(x) = \sqrt{\frac{\sum(x-\bar{x})^2}{n}} \quad (1)$$

$$\text{Upper}_1 = \bar{x} + A.[W(x)] \quad (2)$$

$$\text{Lower}_1 = \bar{x} - A.[W(x)] \quad (3)$$

$$\text{Upper}_2 = \bar{x} + B.[W(x)] \quad (4)$$

$$\text{Lower}_2 = \bar{x} - B.[W(x)] \quad (5)$$

$$JAM = ((\text{Upper}_1 + \text{Lower}_1) - \text{Score}_1) + ((\text{Upper}_2 + \text{Lower}_2) - \text{Score}_2)) \quad (6)$$

Where,
W(x) = Standard deviation,
Upper_1 = Upper Bollinger Band of 1st matrix.
Lower_1 = Lower Bollinger Band of 1st matrix.
Upper_2 = Upper Bollinger Band of 2nd matrix.
Lower_2 = Lower Bollinger Band of 2nd matrix.

C. Algorithm

Input: mov_title ← Movie Title
Output: Dataframe ← 10 movies

1. Import necessary packages
From sklearn.metrics.pairwise import cosine_similarity import math

2. Enter the movie title to find similar movies
mov_title = "Jaws 2"

3. Extract the movie's ID from a pandas DataFrame -> fmovies
mov_id = fmovies[fmovies["original_title"] == mov_title]["imdb_id"]

4. Extract latent feature vectors for the movie
a_1 = np.array (latent_matrix_l_df.loc[mov_id]). reshape (1,-1).
a_2 = np.array (latent_matrix_2_df.loc[mov_id]). reshape (1,-1).

5. Calculate cosine similarity between the movie and all others
Score_1 = cosine_similarity (latent_matrix_l_df, a_1).reshape (-1).
Score_2 = cosine_similarity (latent_matrix_2_df, a_2).reshape (-1).

6. Calculate mean and standard deviation of similarity scores for each model
Mean_1 = np.mean (score_1).
Std_1 = np.std (score_1).
Mean_2 = np. Mean (score_2).
Std_2 = np. Std (score_2).

7. Input the upper and lower factors for Bollinger Bands in the JAM algorithm
X = 2.4
Y = 1.6

Upper_1 = mean_1 + (x*std_1)
Lower_1 = mean_1 - (x*std_1)
Upper_2 = mean_2 + (y*std_2)
Lower_2 = mean_2 - (y*std_2)

8. Calculate hybrid score via JAM formula, using previous scores
Jam = ((upper_1 + lower_1) - score_1) + ((upper_2 + lower_2) - score_2))

9. Construct a Data Frame to store similarity scores
Dict DF = {"content": score_1, "collaborative": score_2, "hybrid": hybrid}
Similar = pd.DataFrame (dictDF, index=latent_matrix_l_df.index)

10. Sort DataFrame in descending order based on hybrid scores
Similar.sort_values ("hybrid", ascending=False, inplace=True).

11. Extract top 10 movie IDs (excluding original movie)
Pred_ids = similar[1:].head(10).index

12. Print titles of recommended movies
For i in pred_ids:
Print (fmovies[fmovies["imdb_id"] == i]["original_title"])

IV. Implementation

The proposed methodology entails the utilization of a collaborative and content-based recommendation matrix, followed by the computation of a hybrid score (JAM) for a given movie. To mitigate the issue of high dimensionality, we employed the SVD module from the sklearn library. Moreover, we utilized the TF-IDF Vectorizer to compute the frequency of keywords in the movie metadata generated during the preprocessing and filtering stages. The Bag of Terms (BoW) model, which relies on the (TF-IDF) term frequency-inverse document frequency, was employed to provide insights on the importance of words in a document.

To ascertain the significance of the features in our dataset, we estimated the variance that they account for, ensuring that they explain at least 25% to 30% of the total variance. This is a crucial step in ensuring that our dataset is well-defined and uniformly distributed across all cases, thereby avoiding sparsity issues that may arise from uneven distribution.

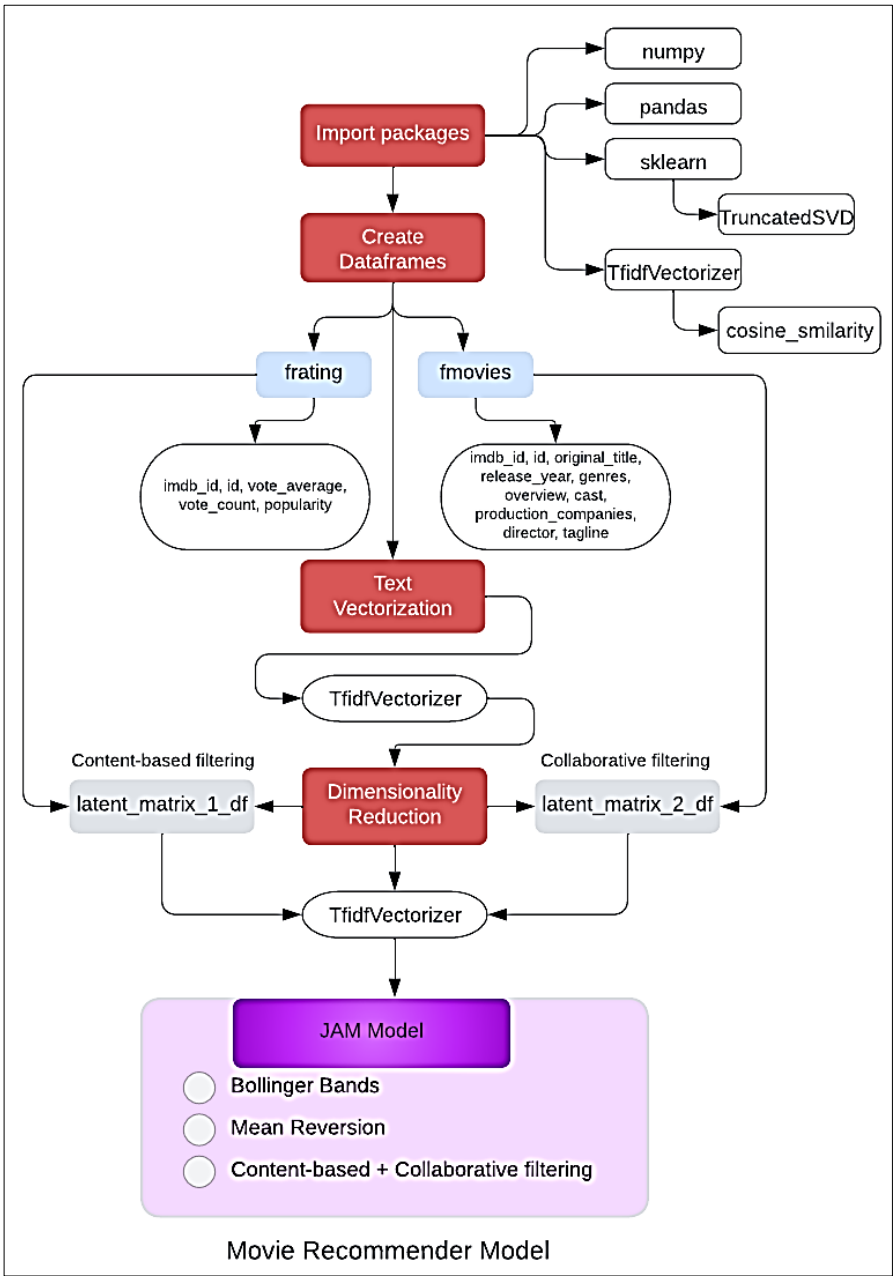


Fig 7: Block Diagram of our Methodology

A. Modules used

We employed the use of various modules in the implementation process, including SVD for dimensionality reduction and TF-IDF Vectorizer to determine the frequency of keywords in the movie metadata. This was achieved using the sklearn library, which provides a range of efficient tools for machine learning and data analysis. By utilizing the Bag of Terms (BoW) model, we were able to determine the relevance of words in a document based on the term frequency inverse document frequency (TF-IDF) approach. It is crucial to ensure that both matrices have the same variance to prevent errors that could arise during matrix operations, which could be attributed to differences in shape or content.

In Python 3.9, the TfidfVectorizer from the scikit-learn library uses the following formula to calculate the TF-IDF (term frequency-inverse document frequency) values for each term in a document:

- n is the total number of documents in the corpus.
- tf(i, j) is the term frequency of term i in document j.
- df(i) is the number of documents containing term i.
- idf(i) is the inverse document frequency of term i.

Then the formula for calculating the TF-IDF value of term i in document j is:

$$\text{TF-IDF}(i, j) = \text{tf}(i, j) * \text{idf}(i) \quad (7)$$

Where,

- tf(i, j) = (number of occurrences of term i in document j) / (total number of terms in document j)
- idf(i) = log(n / df(i))

SVD is a way to factorize a matrix into three matrices: U, Σ, and V, such that:

$$A = U * \Sigma * V^T \quad (8)$$

Where,
- A is the original matrix to be decomposed
- U is an orthogonal matrix of left singular vectors
- Σ is a diagonal matrix of singular values
- V is an orthogonal matrix of right singular vectors
- T denotes the transpose of a matrix

The formula for calculating the SVD factorization of a matrix A is as follows:
- Calculate the eigenvectors and eigenvalues of $A * A^T$ to obtain the left singular vectors and singular values
- Calculate the eigenvectors and eigenvalues of $A^T * A$ to obtain the right singular vectors
- Normalize the left and right singular vectors
- Construct the diagonal matrix of singular values from the square root of the eigenvalues obtained in step 1

The dimensionality reduction is achieved by keeping only the k largest singular values and their corresponding left and right singular vectors. This can be expressed as:

$$A_k = U_k * \Sigma_k * V_k^T \quad (9)$$

Where,
- A_k is the reduced rank approximation of A with k singular values
- U_k is the matrix of the first k columns of U
- Σ_k is the diagonal matrix of the first k singular values
- V_k is the matrix of the first k columns of V

By keeping only the k largest singular values, we can reduce the dimensionality of the original matrix A from m x n to k x k.

B. Dimensionality reduction of our recommendation matrices

Our previous findings showcased the recommendation matrix generated via content-based filtering. Our preprocessed dataset, with 2500 components, displays a variance of around 30%. Utilizing a similar approach, we attained a variance of 70% in the collaborative filtering matrix. While the variances of these matrices differ, it's important to understand that each variance is intrinsic to its respective method and reflects the different methodologies. Discrepancies in variance should not be viewed as an issue, as the matrices are derived from different filtering mechanisms and naturally exhibit different variance levels. Care must be taken when conducting operations or functions on these matrices to avoid potential errors, such as shape or content mismatches. It is imperative to acknowledge and account for these variances to ensure accurate computations.

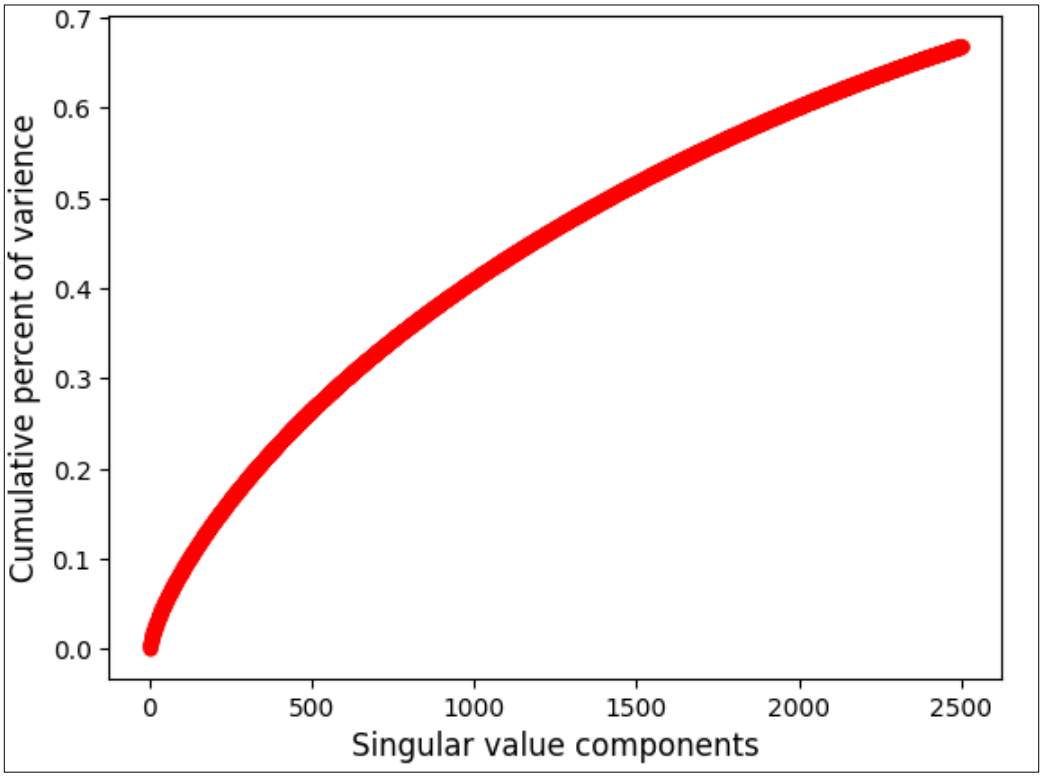


Fig 8a: Variance of content-based Data Frame

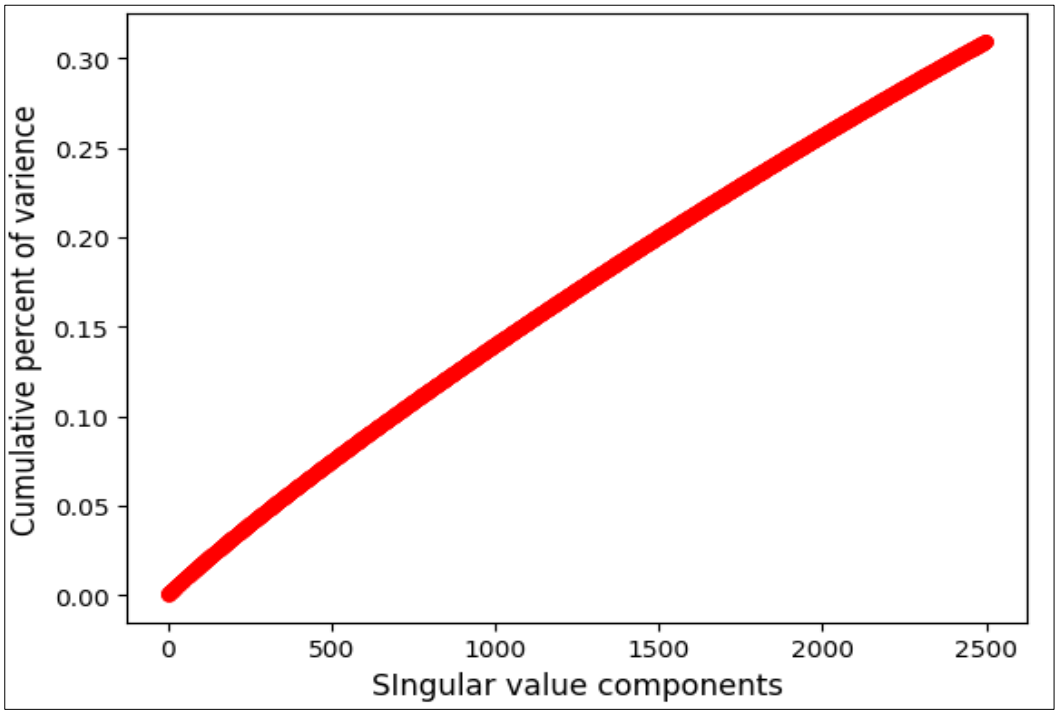


Fig 8b: Variance of collaborative Data Frame

C. Cosine Similarity

The model can be represented by a preferences/rating matrix of order **m x n**, where m is the total number of users (Ur_1, Ur_2, Ur_3,..., Ur_m) and n is the total number of objects (O_1, O_2, O_3,..., O_n) that have been rated by the users. The rating of item 'b' as provided by user 'a' is represented by the cell value $p_{i,j}$ of the matrix. These evaluations may be explicit or implicit.

There are two possible outcomes of collaborative techniques:

i) Prediction of $p_{i,j}$, a number indicating the preference for item 'j' by user 'i'

ii) Recommendation system lists the top 'n' items that are likable by the user

Predicting an item's usefulness for a user is the main goal of the recommender system. A user's level of preference for the movie 'a' is explained in r(e, i). Each item's features provide additional information, i.e., movie ID, actors, director, release Date, genre. The following formula is used to determine an active user's estimated preference for an item 'j':

$$F_{a,j} = \overline{p_a} + z\sum_{i=1}^{n} D_{(a,j)} \times \overline{p_{i,J}} - p_i \quad (10)$$

Where,

D(a, i) is the degree of numerical similarity between the active user and each user 'i' where p_a denotes the mean rating of user 'a', n denotes the number of users in the database with nonzero $p_{i,j}$. The normalizing factor 'z' makes the sum of the absolute values of the weights to one. The similarity between users is calculated using a variety of methods. Some of these techniques are –

i) Pearson Correlation Similarity: Pearson correlation, which has a value between -1 and 1, describes the linear correlation between two vectors. The following formula determines how similar the two vectors a and b are:

$$D_{pearson}(c,d) = \frac{\sum_{i=1}^{n}(\boldsymbol{p_{c,i}}) \cdot (\boldsymbol{p_{d,i}})}{\sqrt{\sum_{i=1}^{n}(\boldsymbol{p_{c,i}})^2 \times (\boldsymbol{p_{d,i}})^2}} \quad (11)$$

To solve the sparsity problem arising from the recommendation of the collaborative filtering method and to consider the consumption pattern changes of the users, we use the Pearson correlation coefficient to classify like-minded users and apply recurrent neural networks to similar user groups.

ii) Cosine Similarity: One of the most widely used statistical techniques for comparing two nonzero real vectors is the cosine. Cosine similarity measures the similarity between two non-zero vectors by computing the cosine of the angle between them. The resulting value ranges from -1 to 1, with 1 indicating perfect similarity and -1 indicating perfect dissimilarity. In n-dimensional space, it seeks out an angle between two vectors and is described as:

$$D_{cosine}(c,d) = \frac{\sum_{i=1}^{n}(p_{c,i}-\overline{p_c}) \times (p_{d,i}-\overline{p_d})}{\sqrt{\sum_{i=1}^{n}(p_{c,i}-\overline{p_c})^2 \times (p_{d,i}-\overline{p_d})^2}} \quad (12)$$

VI. Results: There are several main metrics to evaluate the quality of recommendation systems: prediction accuracy, diversity, coverage, classification accuracy, robustness, real-time, novelty, etc. [4].

To evaluate the performance of the JAM algorithm against other methods, we conducted a comparative analysis involving three additional hybrid algorithms. We presented the results and cluster plots for all methods, with blue clusters representing the scores of the hybrid algorithm, red clusters representing the similarity between hybrid scores and collaborative scores, and green clusters representing the similarity between hybrid scores and content-based scores.
The various formulas under consideration are presented below, with $score_1$ denoting the content-based system score, and $score_2$ denoting the collaborative filtering score: -

A) Simple Average

Simple_average = ($score_1$ +$score_2$)/2

```
10756    Jaws 2
Name: original_title, dtype: object
9806    Jaws
Name: original_title, dtype: object
9603    Jaws: The Revenge
Name: original_title, dtype: object
3822    Sand Sharks
Name: original_title, dtype: object
7840    All That Jazz
Name: original_title, dtype: object
2280    Mega Shark vs. Crocosaurus
Name: original_title, dtype: object
8889    E.T. the Extra-Terrestrial
Name: original_title, dtype: object
1090    The Well
Name: original_title, dtype: object
964    Young Ones
Name: original_title, dtype: object
8576    Flipper
Name: original_title, dtype: object
7851    1941
Name: original_title, dtype: object
```

Fig 9a: Result of first system

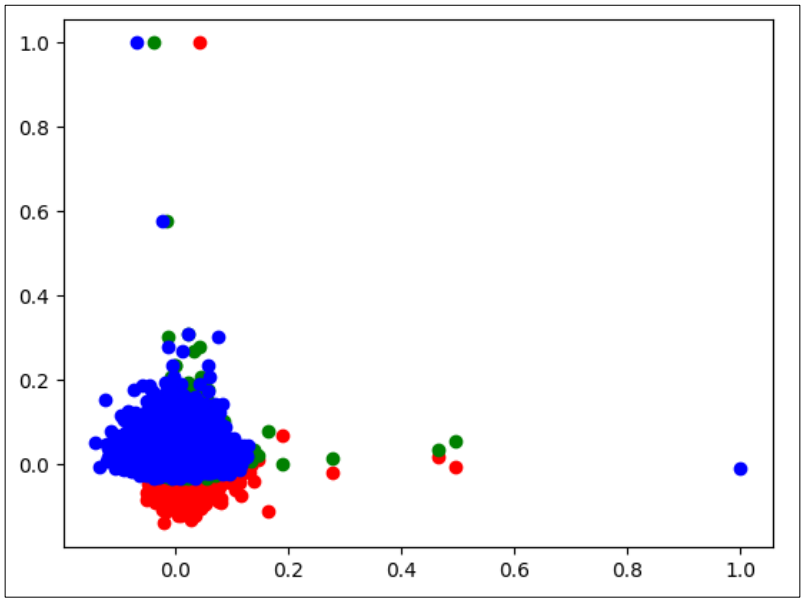


Fig 9b: Cluster plot of first system

B) Std. Dev. style
Hybrid = ((score_1 - mean_1) / std_1 + (score_2 - mean_2) / std_2) / 2.0

```
1134    Seal Team Eight: Behind Enemy Lines
Name: original_title, dtype: object
9806    Jaws
Name: original_title, dtype: object
9603    Jaws: The Revenge
Name: original_title, dtype: object
3822    Sand Sharks
Name: original_title, dtype: object
7840    All That Jazz
Name: original_title, dtype: object
2280    Mega Shark vs. Crocosaurus
Name: original_title, dtype: object
1090    The Well
Name: original_title, dtype: object
8889    E.T. the Extra-Terrestrial
Name: original_title, dtype: object
7851    1941
Name: original_title, dtype: object
964     Young Ones
Name: original_title, dtype: object
```

Fig 10a: Result of second system

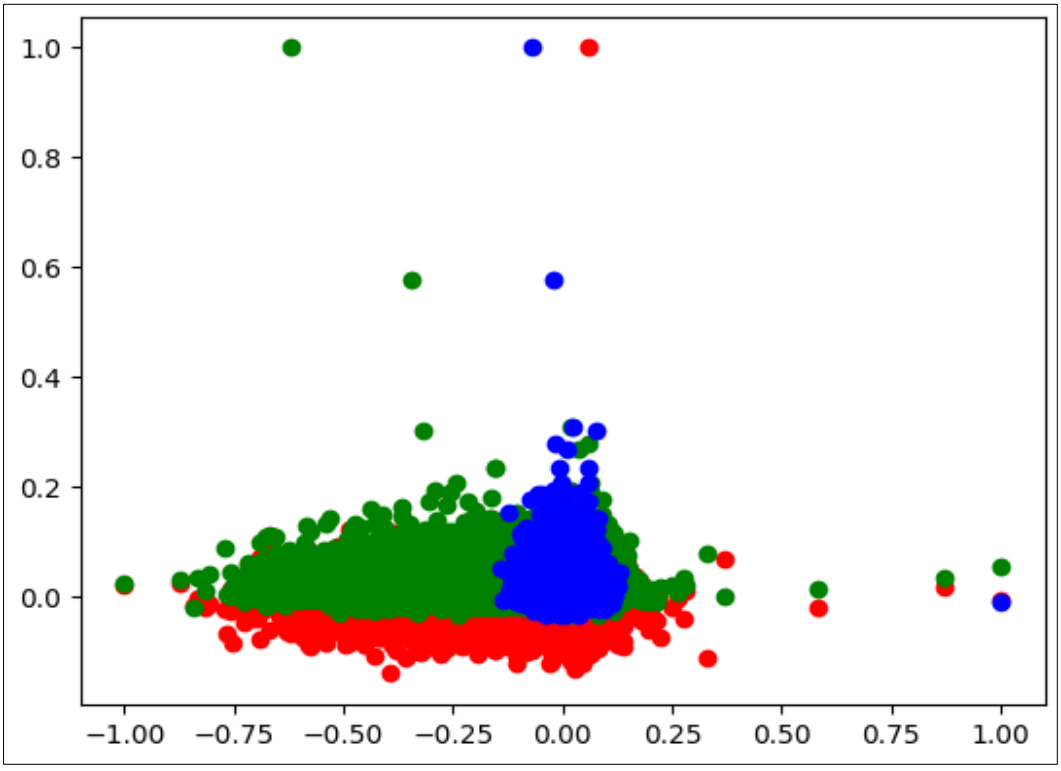


Fig 10b: Cluster plot of second system

C) Exponential formula

Hybrid = ((upper_1+lower_1)/2) + ((upper_2 + lower_2)/2)^3

```
7624    Bury My Heart At Wounded Knee
Name: original_title, dtype: object
7641    White Noise 2: The Light
Name: original_title, dtype: object
7640    FrontiÃ¨re(s)
Name: original_title, dtype: object
7639    Prey
Name: original_title, dtype: object
7638    The Last Legion
Name: original_title, dtype: object
7637    The Good Night
Name: original_title, dtype: object
7636    How to Rob a Bank
Name: original_title, dtype: object
7634    Interview
Name: original_title, dtype: object
7633    Silk
Name: original_title, dtype: object
7631    The Dukes of Hazzard: The Beginning
Name: original_title, dtype: object
```

Fig 11a: Result of third system

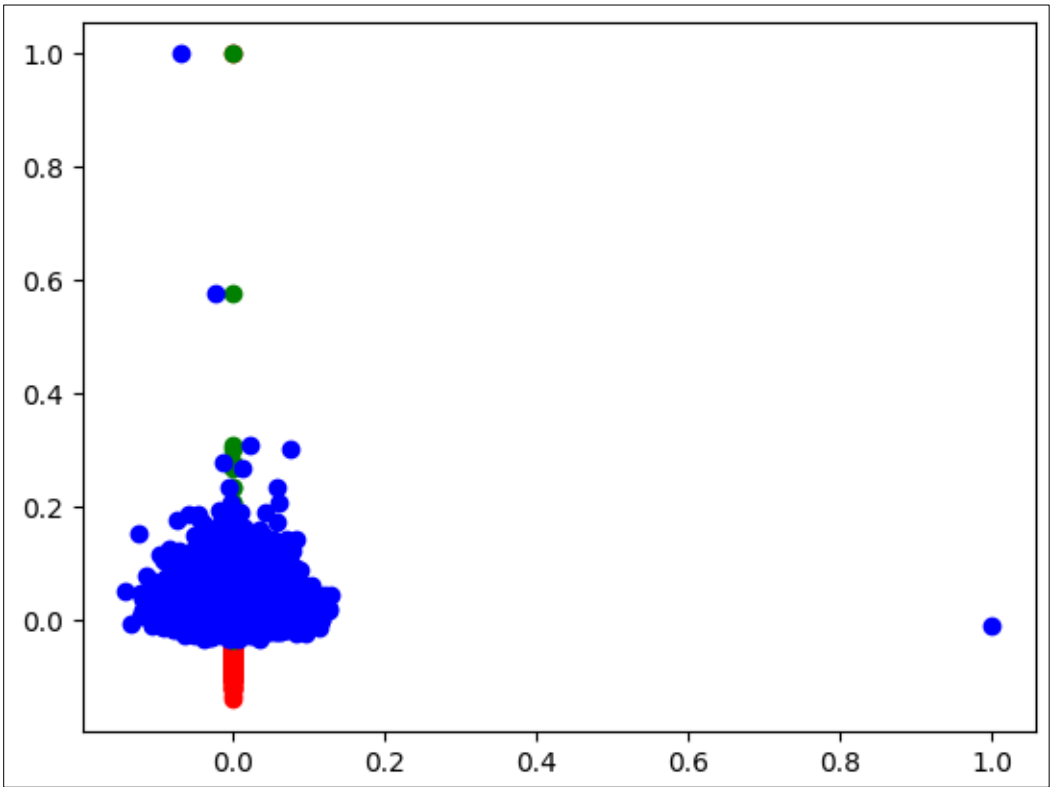


Fig 11b: Cluster plot of third system

D) JAM formula

JAM = ((upper$_1$+lower$_1$) – score$_1$) + ((upper$_2$ + lower$_2$) – score$_2$)

```
980     The Humbling
Name: original_title, dtype: object
5787     The Face of Love
Name: original_title, dtype: object
5693     The Invisible Woman
Name: original_title, dtype: object
3166     Winged Creatures
Name: original_title, dtype: object
3723     Wild Bill
Name: original_title, dtype: object
5766     The Look of Love
Name: original_title, dtype: object
3973     Frida
Name: original_title, dtype: object
3201     Lost in Austen
Name: original_title, dtype: object
3541     Marley & Me: The Puppy Years
Name: original_title, dtype: object
4968     Bad Santa
Name: original_title, dtype: object
```

Fig 12a: Results of JAM

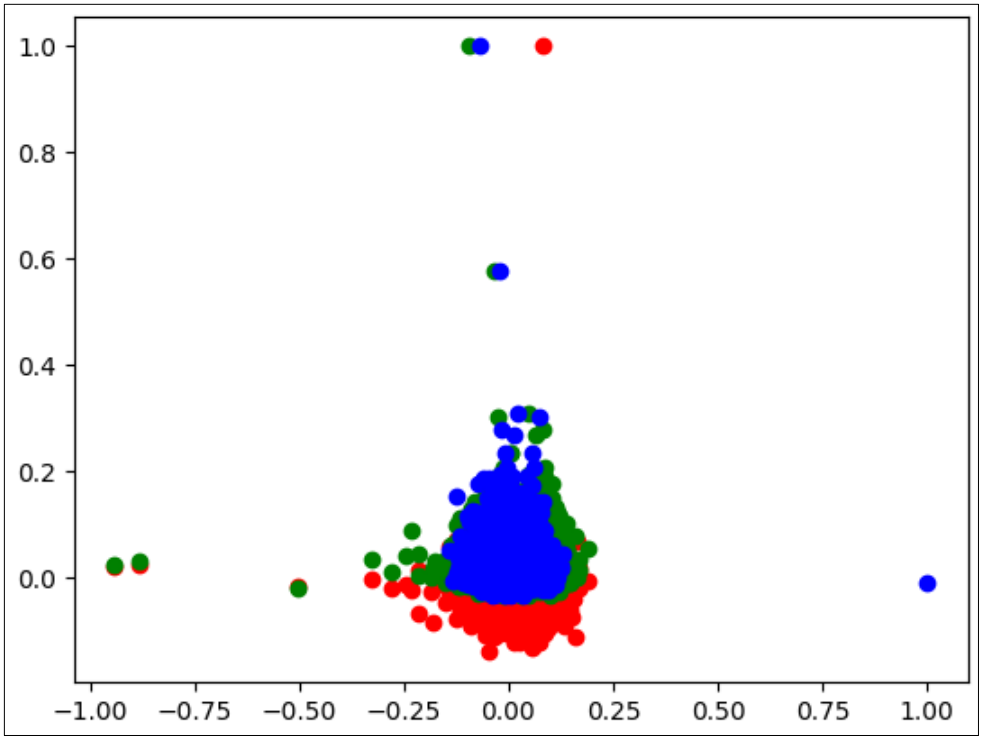


Fig 12b: Cluster plot of JAM

From figures 8, 9, 10 and 11, the inferences we can come to are

- Clustering in JAM algorithm is the best as it overlaps nearly every facet of collaborative and content-based scores.
- Exponential systems lead to very weak recommender systems as they only have a 1-D frame of hybrid recommendation.
- The std. dev. style system has a very broad spectrum and the recommendations thus provided would be too weak to even be deemed valid.
- The simple average system gives good results but still falls short to our JAM system.
- JAM recommendations are very varied in terms of genre.
- Most of the other systems recommended either a sequel or prequel to the movie inputted, which is a very basic form of recommendation. This drawback cannot be found in JAM.

Conclusion

The proposed system addresses some of the limitations of traditional recommendation systems, such as the over-reliance on user preferences and inability to recommend new or overlooked items. By incorporating mean reversion, the system can identify undervalued movies and recommend them to users who are likely to enjoy them. Furthermore, the use of a hybrid approach allows the system to combine the strengths of both collaborative and content-based filtering methods, resulting in more accurate and personalized recommendations.

Overall, the proposed movie recommendation system offers a robust and efficient solution to the challenge of providing personalized movie recommendations to users.

Future Direction

Future research could explore the integration of additional features and techniques, such as natural language processing and deep learning, to further improve the accuracy and effectiveness of the system. The criterion of recommending movies solely based on similar content-elements may not be a decisive factor for all users, leading to inadequate movie suggestions generated by content-based filtering during a cold start problem in collaborative-filtering.

To enhance the user experience, we propose the development of a dataset with embedded compounded tags, which would offer more comprehensive movie recommendations. Additionally, to cater to users who highly regard critics' opinions, we suggest incorporating a critic-specific score into the recommender system datasets. To further augment the platform, we also recommend the integration of critics' reviews, with appropriate compensation arrangements for the critics involved. Basuroy *et. al* examine whether critics act as influencers or predictors of the movie-going public's decision and find that both positive and negative reviews are correlated with weekly box office revenue over an eight-week period. The authors also find that the impact of negative reviews decreases over time, indicating the influence of critics. Additionally, the study reveals a negativity bias in film reviews, where negative reviews have a greater impact on box office performance than positive reviews, particularly during the first week of a film's release. The authors further investigate the moderating effects of stars and budgets on critical reviews and find that popular stars and big budgets enhance box office revenue for films with more negative critical reviews than positive reviews [15].

References

1. Hwang S, Park E. Movie Recommendation Systems Using Actor-Based Matrix Computations in South Korea. IEEE Transactions on Computational Social Systems. 2021 Oct 11;9(5):1387-93. DOI: 10.1109/TCSS.2021.3117885.
2. Feng X, Hu J, Zhu X. "Machine Learning Based Personalized Movie Research and Implementation of Recommendation System," 2022 International Conference on Culture-Oriented Science and Technology (CoST), Lanzhou, China, 2022, 74-78. Doi: 10.1109/CoST57098.2022.00025.
3. Xu Q, Han J. "The Construction of Movie Recommendation System Based on Python," IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), Chongqing, China; c2021. p. 208-212. DOI: 10.1109/ICIBA52610.2021.9687872.
4. Chen Q, Qin J. "Research and implementation of movie recommendation system based on deep learning," 2021 IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI), Fuzhou, China; c2021. p. 225-228. DOI: 10.1109/CEI52496.2021.9574461.
5. Zan Wang, Xue Yu, Nan Feng, Zhenhua Wang, An improved collaborative movie recommendation system using computational intelligence, Journal of Visual Languages & Computing. 2014;25(6):667-675. ISSN 1045-926X,
6. Lund J, Ng YK. Movie recommendations using the deep learning approach. In2018 IEEE international conference on information reuse and integration (IRI) 2018 Jul 6 pp 47-54. IEEE. doi: 10.1109/IRI.2018.00015.
7. Halder S, Sarkar AJ, Lee YK. "Movie Recommendation System Based on Movie Swarm," 2012 Second International Conference on Cloud and Green Computing, Xiangtan, China; c2012. p. 804-809, DOI: 10.1109/CGC.2012.121.
8. Wang H, Lou N, Chao Z. "A Personalized Movie Recommendation System based on LSTM-CNN," 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Taiyuan, China; c2020. p. 485-490. DOI: 10.1109/MLBDBI51377.2020.00102.
9. Kim M, Jeon S, Shin H, Choi W, Chung H, Nah Y. "Movie Recommendation based on User Similarity of Consumption Pattern Change," 2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), Sardinia, Italy; c2019. p. 317-319, DOI: 10.1109/AIKE.2019.00064.
10. Kim HN, El-Saddik A, Jo GS. "Collaborative error-reflected models for cold-start recommender systems," Decis. Support Syst. Jun. 2011;51(3):519–531.
11. Choi SM, Ko SK, Han YS. A movie recommendation algorithm based on genre correlations. Expert Systems with Applications. 2012 Jul 1;39(9):8079-85.

12. Gui-Rong Xue, Chenxi Lin, Qiang Yang, WenSi Xi, Hua-Jun Zeng, Yong Yu, *et al*. Scalable collaborative filtering using cluster-based smoothing. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '05). Association for Computing Machinery, New York, NY, USA; c2005. p. 114-121.
13. Pazzani MJ. A framework for collaborative, content-based and demographic filtering. Artificial intelligence review. 1999 Dec;13:393-408.
14. Deldjoo Y, Elahi M, Cremonesi P, Garzotto F, Piazzolla P, Quadrana M. Content-based video recommendation system based on stylistic visual features. Journal on Data Semantics. 2016;5(2):99-113.
15. Basuroy S, Chatterjee S, Ravid SA. How critical are critical reviews? The box office effects of film critics, star power, and budgets. Journal of marketing. 2003 Oct;67(4):103-17. https://doi.org/10.1509/jmkg.67.4.103.18692
16. Deuk Hee Park, Hyea Kyeong Kim, Young Choi Il, Jae Kyeong Kim. A literature review and classification of recommender systems research, Expert Systems with Applications, 2012 Sep 1;39(11):10059-72.. ISSN 0957-4174.
17. Georgiou O, Tsapatsoulis N. Improving the Scalability of Recommender Systems by Clustering Using Genetic Algorithms. InICANN (1); c2010. p. 442-449.
18. Arsalan T, Chishty BA, Ghouri S, Ansari NU. Comparison of volatility and mean reversion among developed, developing and emerging countries. Journal of Economic and Administrative Sciences. 2022 Sep 22(ahead-of-print). https://doi.org/10.1108/JEAS-01-2022-0009
19. https://www.investopedia.com/terms/b/bollingerbands.asp
20. https://www.fidelity.com/learning-center/trading-investing/technical-analysis/technical-indicator-guide/bollinger-bands#:~:text=Bollinger%20Bands%20are%20envelopes%20plotted,Period%20and%20Standard%20Deviations%2C%20Std Dev
21. https://www.bollingerbands.com/bollinger-bands