Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/180504

Next-Generation Movie Recommenders: Leveraging Hybrid Deep Learning for Enhanced Personalization

Deema Mohammed Alsekait[1,*], *Ahmed Younes Shdefat*[2,*], *Nour Mostafa*[2,*], *Alaa Mohamed Mohamed Hamdy*[3], *Hanaa Fathi*[4], *and Diaa Salama AbdElminaam*[4,5,6,7]

[1] Department of Computer Science and Information Technology, Applied College, Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia
[2] College of Engineering and Technology, American University of the Middle East, Egaila 54200, Kuwait
[3] Faculty of Engineering, Ain Shams University, Cairo, Egypt
[4] Applied Science Research Center, Applied Science Private University, Amman, Jordan
[5] Faculty of Computers and Artificial Intelligence, Benha University, Benha, Egypt
[6] Faculty of Computer Science, Misr International University, Cairo, Egypt
[7] MEU Research Unit, Middle East University, Amman, Jordan

Received: 2 Feb. 2024, Revised: 23 Mar. 2023, Accepted: 24 May 2024
Published online: 1 Sep. 2024

Abstract: This study aims to advance movie recommendation systems by integrating and comparing multiple machine learning and deep learning algorithms, specifically focusing on Contact-Based Filtering and Content Filtering techniques such as TF-IDF, Decision Trees, K-Nearest Neighbors, Singular Value Decomposition (SVD), Neural Collaborative Filtering (NCF), ALS, and the AUTOENCODER. We comprehensively evaluated these methodologies using three distinct datasets—MovieLens Dataset, The Movies Dataset, and TMDB Movie Dataset. The results indicate that different algorithms excel depending on the dataset characteristics. SVD shows superior performance on the MovieLens Dataset, AUTOENCODER excels in The Movies Dataset, and a hybrid approach proves most effective on the TMDB Movie Dataset. The study emphasizes the importance of considering dataset specificities when selecting recommendation algorithms to optimize system performance. Our findings contribute to developing more personalized, accurate movie recommendation systems and highlight the potential of hybrid approaches in addressing diverse user preferences and content complexities. Integrating various machine learning techniques significantly enhances the adaptability and accuracy of recommendation systems, presenting a substantial advancement in personalized content delivery. We leverage three datasets (MovieLens Dataset, The Movies Dataset, and TMDB Movie Dataset) for comprehensive evaluation and comparison. In our analyses, SVD demonstrated superior performance in the MovieLens 20M Dataset, while transitioning to The Movies Dataset highlighted AUTOENCODER's exceptional performance, and the TMDB 5000 Movie Dataset showcased the hybrid recommender as the most effective. The study emphasizes the importance of considering specific dataset characteristics for optimal recommendation algorithm selection. Integrating Contact-Based Filtering, Content Filtering, and Neural Collaborative Filtering significantly improves recommendation system accuracy. While Singular Value Decomposition excels, further investigation considering dataset characteristics and user preferences is deemed essential for algorithm selection in diverse scenarios.

Keywords: Collaborative Filtering, Machine Learning Algorithms, Deep Learning Models, User Behavior Analysis , Predictive Analytics , Data Preprocessing Techniques Personalization Strategies

1 Introduction

The integration of machine learning techniques, specifically Contact-Based Filtering and Content Filtering, within Recommendation Systems represents a pivotal advancement in the realm of personalized movie suggestions. This transformative integration is crucial in responding to the dynamic and diverse preferences of users, fundamentally enhancing their overall movie-watching experience. By leveraging Contact-Based Filtering, these systems can analyze user interactions and similarities, ensuring that recommendations are not only tailored to individual preferences but also consider the

* Corresponding author e-mail: Dmalsekait@pnu.edu.sa, ahmed.shdefat@aum.edu.kw, nour.moustafa@aum.edu.kw

viewing patterns of like-minded users. Simultaneously, Content Filtering allows for a deeper understanding of the intrinsic characteristics of movies, enabling the system to recommend content based on thematic elements, genres, or specific attributes that align with a user's taste. This innovative approach not only addresses the challenges of information overload but also ensures that users are presented with relevant and enjoyable movie suggestions, ushering in a new era of personalized and enriched cinematic experiences. As the film industry continues to diversify, the importance of fine-tuning recommendation systems becomes increasingly evident, contributing significantly to the democratization of content consumption and fostering a more engaging and satisfying interaction between users and the vast landscape of cinematic offerings.

Despite remarkable advancements in recommendation systems, a critical gap persists in the comprehensive evaluation of diverse recommendation algorithms. The sheer diversity and complexity of modern algorithms pose challenges in determining their effectiveness across various contexts and datasets. This gap hinders the ability to make well-informed decisions when selecting algorithms for specific applications, leading to potential suboptimal choices in system design. Necessitating further research, this gap underscores the urgency to delve deeper into algorithmic evaluation, exploring nuances that impact performance across different scenarios. Bridging this critical gap is essential to advance the field, enabling practitioners and researchers to make informed choices that optimize recommendation system outcomes and ultimately enhance user satisfaction. The ever-evolving landscape of user preferences and content offerings necessitates a thorough understanding of algorithmic strengths and weaknesses, making the imperative to address this gap even more pressing in the dynamic domain of recommendation systems.

Machine learning and deep learning play pivotal roles in shaping the landscape of recommendation systems, contributing to the evolution of personalized content delivery. Contact-Based Filtering and Content Filtering methodologies represent two distinct approaches employed in these systems. The former analyzes user interactions and similarities, utilizing techniques like TF-IDF (Term Frequency-Inverse Document Frequency), Decision Trees, and K-Nearest Neighbors to identify patterns and recommend items based on user behaviors. Content Filtering, on the other hand, understands the intrinsic characteristics of items, employing methods like Singular Value Decomposition (SVD) to provide recommendations based on thematic elements and genre affinity. The integration of Neural Collaborative Filtering (NCF) adds a layer of sophistication, leveraging deep learning architectures to comprehend intricate user-movie interactions and predict preferences accurately. This amalgamation of traditional machine learning and advanced deep learning techniques reflects the continuous efforts to enhance the efficiency and personalization capabilities of recommendation systems in an ever-expanding and dynamic digital content landscape. .

The problem addressed in this paper stems from the complexity and inefficiency of existing movie recommendation systems in navigating the vast and diverse landscape of cinematic content. Traditional recommendation systems often struggle to adequately capture and respond to the nuanced preferences of individual users, leading to a gap in delivering truly personalized and satisfying viewing suggestions. This challenge is exacerbated by the dynamic nature of user preferences and the continuous expansion of available content, which demand more sophisticated and adaptable recommendation solutions.

The objective of this research is to significantly enhance the efficacy of movie recommendation systems by leveraging a range of both machine learning and deep learning techniques. By systematically evaluating and integrating methods such as Contact-Based Filtering, Content Filtering, TF-IDF, Decision Trees, K-Nearest Neighbors, Singular Value Decomposition, Neural Collaborative Filtering, ALS, and the AUTOENCODER, the study aims to identify optimal strategies for different types of data. This approach is intended to improve the accuracy, personalization, and user satisfaction of recommendations, ensuring that they are both relevant and engaging based on individual user profiles and preferences.

The overarching objective of our study is to elevate the performance of movie recommendation systems by delving into the realm of Contact-Based Filtering and Content Filtering methodologies. This exploration encompasses a diverse array of techniques, such as TF-IDF, Decision Trees, K-Nearest Neighbors, Singular Value Decomposition (SVD), Neural Collaborative Filtering (NCF), ALS, and the AUTOENCODER. To comprehensively evaluate and compare these methodologies, we leverage three distinct datasets: the MovieLens Dataset, The Movies Dataset, and TMDB Movie Dataset. Our approach aims to provide a thorough understanding of each model's efficacy across varied datasets, enabling us to tailor our recommendation system for optimal performance in different scenarios. Through this rigorous evaluation, we seek to advance the capabilities of movie recommendation systems, ensuring enhanced accuracy and relevance for diverse user preferences and content landscapes.

This research makes several significant contributions to the field of movie recommendation systems, which include:

– **Comprehensive Methodology:** Implementation of a robust analysis comparing various machine learning and deep learning techniques for improving movie recommendation systems. The methodologies employed include Contact-Based Filtering, Content Filtering, TF-IDF, Decision Trees, K-Nearest Neighbors, Singular Value Decomposition (SVD),

Neural Collaborative Filtering (NCF), ALS, and AUTOENCODER.

- **Dataset Evaluation:** Utilization of three distinct datasets—MovieLens, The Movies Dataset, and TMDB Movie Dataset—to evaluate the performance of each algorithm. This diverse dataset usage provides a thorough understanding of each model's efficacy across varied scenarios, highlighting the importance of dataset-specific algorithm selection.
- **Algorithm Performance Analysis:** Detailed performance analysis where SVD was found to excel with the MovieLens 20M Dataset, AUTOENCODER showed exceptional results on The Movies Dataset, and a hybrid recommendation approach was most effective for the TMDB 5000 Movie Dataset.
- **Enhancement of Recommendation Systems:** The study enhances the adaptability and accuracy of movie recommendation systems by effectively integrating and balancing multiple recommendation techniques. This integration caters to diverse user preferences and content complexities, thus improving personalization and user satisfaction.
- **Practical Implications:** The findings provide actionable insights for developers and researchers in the entertainment industry to refine their recommendation engines, ensuring that users receive more accurate and relevant content suggestions.

These contributions collectively advance the capability of movie recommendation systems to offer more personalized and user-centric suggestions, addressing both the scalability and complexity challenges prevalent in current systems.

The rest of the paper is organized as follows: In Section 2, a comprehensive overview of related work is presented. Section 3 introduces the proposed methodology and framework for the Movie Recommendation System, delineating the six significant steps involved in its implementation. Subsequently, Section 3.1 focuses on the critical phase of data collection. In Section 3.2, the paper delves into the intricacies of data preprocessing. Following this, Section 3.3 provides insights into the data splitting process, showcasing tailored strategies for different algorithms. Section 3.4 dives into the optimization of hyperparameters for both Machine Learning (ML) and Deep Learning (DL) models. Moving forward, Section 3.5 explores the recommendation models based on both ML and DL algorithms. The subsequent section, Section 3.6, highlights the pivotal role of performance metrics in evaluating the efficacy of the recommendation systems. In Section 4, the experimental results and discussions unfold, presenting a detailed analysis of the performance of recommendation systems. Finally, Section 5 offers a conclusion and outlines avenues for future work.

2 related work

Exploring the current landscape of research and advancements in the realm of movie recommendation systems. This comprehensive examination focuses on two pivotal methodologies—content-based filtering and collaborative filtering—while also delving into the integration of machine learning and deep learning techniques. Through a judicious synthesis of existing literature, this section endeavors to trace the evolutionary trajectory of movie recommendation paradigms, spotlighting key findings and technological advancements that collectively define the contemporary landscape of personalized content recommendation.

In [1] , the author presented a hybrid approach in this research paper with the goal of improving the scalability, quality, and accuracy of movie recommendation systems. This method uses a genetic algorithm and Support Vector Machine as a classifier to combine collaborative filtering with content-based filtering. Using three different MovieLens datasets, the study compares the suggested hybrid approach to pure content-based and collaborative filtering techniques. The results show that the hybrid approach reduces computing time and performs better than the pure approaches in terms of movie recommendation quality, scalability, and accuracy. It should be noted, though, that the suggested approach requires more memory than the pure approaches. The research also suggests potential future work, including the consideration of user age in movie preferences and addressing the memory requirements of the proposed approach.

In [2], the authors of this paper suggests a hybrid recommendation system to enhance the prediction and recommendation accuracy of films by combining non-personalized (Bayesian Estimation) and content-based (TF-IRF approach with Cosine Similarity) techniques. The two data mining methods that make up the system architecture are Inverse rating frequency (Cosine Similarity) and Bayesian Estimation, which are combined in the business logic section. The movie metadata is gathered and categorised in the database as a training set, and Django is the web application framework used to combine all the features. With a reduction in web page loading times from 0.35 to 0.56 to 0.32 to 0.41 seconds, the results demonstrate the effectiveness of the suggested strategy.The proposed hybrid method is more efficient in all aspects when compared to the old conventional methods, which are bounded by certain limitations.

In [4] ,the authors of the study introduces a novel collaborative filtering metric for recommender systems that takes into account user voting patterns and the summation of ratings from users who are similar to each other. By using mean squared differences and the Jaccard measure, it outperforms conventional metrics like Pearson correlation. Comparing the MovieLens and NetFlix databases to Pearson correlation, the experiments showed

notable improvements in mean absolute error, coverage, and percentage of perfect predictions. When the new metric was applied to the FilmAffinity database, it did not demonstrate any improvement, most likely because the votes in that dataset ranged from 1 to 10 stars.

In [3], the authors presents a hybrid movie recommender system that combines collaborative filtering with content-based filtering using neural network classifiers. The system automatically retrieves contributors' details and movie summaries using a Java parser. By training on movie features like stars, genres, and synopses, neural networks are able to achieve high success rates in precise recommendation. An overall percentage of successful recommendations of 82% is obtained by fuzzy aggregation of filtering outputs, which performs better than classical Boolean aggregation. By increasing the contribution of each individual criterion, the system's performance could be further improved. Along with discussing scalability issues, the paper makes recommendations for future research directions, including enhancing the selection of films for users with limited viewing history and incorporating semantic relations in synopsis representation.

In [5], with an emphasis on preference-based filtering through the integration of content-based and collaborative filtering principles, the authors present a novel strategy for online decision support. Preference modelling and machine learning are integrated to enable scalability with a large user base, and algorithms supporting the ability to explain and justify recommendations are presented. Key characteristics of the "film-conseil" system, which applies the suggested methodology to online movie recommendation tasks, are also showcased. The study addresses the drawbacks of exclusive collaborative filtering and promotes the combination of content-based techniques and pure collaborative filtering. Along with encouraging users to actively rate items, it also tackles the issue of the system's recommendation engine's need for explanation.The suggested architecture for recommender systems combines machine learning and preference-based search, emphasising the interpretation of recommendations, encouraging user evaluations, and responding to user input. In order to enable users to visualise their inferred profiles and challenge rules that the system has learned, the paper discusses the use of decision trees for preference profiling and active learning. In order to expand the possible applications of the suggested method, it also addresses the necessity for tools to extract the content of semi-structured documents.

In [6], the authors of this paper, machine learning models are used to develop and implement a movie recommendation system. XGBoost, Matrix Factorization SVD, Surprise Baseline, Surprise KNNBaseline, and Matrix Factorization SVDpp are among the models that were employed. With a test RMSE of 1.0675, SVDpp proved to be the most effective model. Along with investigating user-item and item-item similarity matrices, the study tackles the cold start issue. A thorough literature review on collaborative filtering techniques is also provided, and the paper highlights the significance of recommendation systems in today's hectic world.

In [7], The authors use collaborative filtering in the Spark engine to present a comparative analysis of movie recommendation systems. The Alternating Least Squares (ALS) algorithm is the main topic of the study, and it is compared to other algorithms. The findings indicate that the ALS algorithm outperformed the Singular Value Decomposition, K-Nearest Neighbour, and Normal Predictor algorithms in terms of Root Mean Square Error (RMSE) values. The most accurate model was created using the ALS algorithm, an 80-20 dataset, a regularisation parameter of 0.1, and 20 iterations.

In [8], The paper explores various models for movie recommendation, including Genre-Based Recommendation, Pearson Correlation Coefficient-Based Recommendation, Cosine Similarity-Based Recommendation, KNN-Based Recommendation with Cosine as the metric, K-Means Clustering-Based Recommendation, Latent Matrix using TFIDF & SVD for Content-Based Filtering, Latent Matrix using TFIDF & SVD for Collaborative Filtering, and Surprise Library (utilizing KNN Basic) Based Recommendations. Through evaluation on the Movie Lens dataset, it is revealed that the collaborative filtering approach, specifically involving TFIDF and SVD, yields optimal results. Additionally, the study underscores the efficacy of the Surprise library in recommending movies through the KNN algorithm. By employing machine learning techniques to create clusters based on movie genres, the research presents promising outcomes, offering valuable insights into the performance and applicability of diverse recommendation system technologies.

In [9], The paper by Behera et al presents a hybrid model for movie recommendation systems that combines content-based K-nearest neighbors with a restricted Boltzmann machine. Through experiments conducted on MovieLens benchmark datasets, the authors showcase that the proposed model attains superior accuracy and efficiency in providing movie recommendations to active users. By harnessing the strengths of both content-based and collaborative filtering techniques, the hybrid model surpasses the performance of individual methods, effectively addressing challenges associated with sparse datasets and rating prediction. This innovative approach contributes to elevating the overall efficacy of movie recommendation systems, representing a promising advancement in the realm of recommendation systems.

In [10], The paper titled "Movie Recommendation System Using Semi-Supervised Learning" introduces a movie recommendation system that leverages semi-supervised learning. Utilizing a collaborative-based filtering approach, the system predicts movie ratings for users and achieves an estimated prediction of 2.58 for movie ID 302. The evaluation of recommendation accuracy employs metrics like Mean Absolute Error (MAE), Percentage Error, Mean Squared Error (MSE),

and Root Mean Square Error (RMSE). The obtained results highlight the effectiveness of the recommendation approaches, offering insights into the performance of both simple recommender and content-based filtering methods. Furthermore, the paper explores potential future enhancements, including deploying the application in the cloud and integrating deep learning for dynamic recommendations.

In [11], The paper authored by Roy and Ludwig, titled "Genre based hybrid filtering for movie recommendation engine," introduces a collaborative filtering algorithm based on movie genres to tackle the Cold-Start problem in movie recommendations. In addressing this challenge, the proposed algorithm integrates users' viewing history with movie genre information to provide recommendations. It exhibits enhanced accuracy in predicting user ratings, particularly in scenarios with limited user-item interactions. Comparative evaluations against traditional collaborative filtering methods such as Item-Item Collaborative Filtering (IIHF) and User-User Collaborative Filtering (UUCF) reveal superior performance in terms of Mean Absolute Error (MAE) and Mean Squared Error (MSE) across various levels of data sparsity. The outcomes underscore the efficacy of the Genre-Similarity Hybrid Filtering (GSHF) approach, positioning it as a promising solution for addressing cold-start challenges in movie recommendation systems.

In [12], The paper titled "A Comprehensive Survey on Cross-Domain Recommendation: Toward Context-Aware Recommendation" delivers an exhaustive examination of cross-domain recommendation systems, with a specific emphasis on context-aware recommendation techniques. In this survey, the authors scrutinize a range of models and algorithms applied in cross-domain recommendation, encompassing matrix factorization, deep learning, and transfer learning. Addressing the challenges and opportunities within this domain, the paper presents in-depth analyses of results and accuracies achieved by diverse models in cross-domain recommendation tasks, emphasizing the effectiveness of context-aware approaches. Notably, the document explores the performance of matrix factorization in capturing cross-domain correlations and underscores the heightened accuracy of deep learning models in managing contextual information. In summary, the survey provides valuable insights into state-of-the-art techniques and their efficacy in the realm of cross-domain recommendation.

In [13], The paper assesses two collaborative filtering techniques, namely matrix factorization and user-based collaborative filtering (UBCF) utilizing a cosine similarity function, employing the Movielens dataset comprising one million ratings. Evaluation is conducted based on the Root Mean Square Error (RMSE) for both the entire dataset and various partitions categorized by age, genre, or date of rating. The findings reveal that the RMSE for the complete dataset is lower than that of each individual partition. Introducing a novel hybrid technique that incorporates age, genre, and date into the definition of the cosine similarity function, the study demonstrates its superior performance over traditional UBCF. The adapted cosine similarity function also outperforms conventional UBCF, while the RMSE of matrix factorization exhibits variations contingent on the training set size and dataset ratings.

In [14], The paper "Machine Learning Model for Movie Recommendation System" by M. Chenna Keshava, P. Narendra Reddy, S. Srinivasulu, and B. Dinesh Naik, published in the International Journal of Engineering Research & Technology, Vol. 9, Issue 04, April 2020, presents a comprehensive study on enhancing the CineMatch algorithm by incorporating collaborative filtering techniques. The authors evaluated several models including XGBoost, BaselineOnly, KNNBaseline, Matrix Factorization SVD, and SVDpp. Among these, the SVDpp model demonstrated the best performance with a test RMSE of 1.0675. The study also outlines future enhancements such as hyperparameter tuning and utilizing cloud resources for training on the entire dataset to further improve the recommendation system.

In [15], The paper assesses three types of recommender systems, namely content-based, collaborative filtering, and hybrid. The authors specifically concentrate on collaborative filtering, conducting a comparison between user-based collaborative filtering utilizing modified cosine functions and a novel hybrid technique that incorporates age, genre, and date into the cosine similarity function. The evaluation utilizes the Movielens dataset, encompassing 100,000 and 1 million ratings. The findings reveal that the RMSE of the complete dataset is lower than that of individual partitions, and the newly introduced hybrid technique surpasses traditional collaborative filtering methods, particularly in scenarios with small training sets. The authors also highlight the additive nature of the innovative technique, emphasizing its capacity to maximize the contribution of each partition without disregarding others.

In [4], The paper presents a hybrid movie recommender system employing neural networks that integrates content-based and collaborative filtering techniques. Neural networks are utilized for content-based filtering, evaluating movie genres, contributors, and synopses, while collaborative filtering is implemented using the Pearson formula to identify correlations among users. The amalgamation of filtering outcomes is accomplished through fuzzy logic operators. Experimental results showcase an overall percentage of successful recommendations at 82%, with a mean success rate per user reaching 81.8%. Precision and recall metrics are employed to assess the system's precision, revealing a success rate of 78.5% for content-based filtering alone.

In [16], The paper introduces an enhanced approach for a movie recommendation system through a hybrid model that merges content-based filtering and collaborative filtering. The model incorporates Support Vector Machine as a classifier and utilizes a genetic

algorithm to enhance the quality, accuracy, and scalability of the movie recommendation system. The proposed methodology is implemented and assessed against existing standalone approaches using three distinct Movie Lens datasets. The results indicate that the hybrid approach proposed in the paper improves the accuracy, scalability, and quality of the movie recommendation system compared to the standalone approaches. Additionally, the computing time of the proposed approach is found to be shorter, although it comes with a higher memory requirement compared to existing standalone approaches.

In [17], The paper titled "Movie Recommender System Using Machine Learning Algorithms" explores the creation of a movie recommender system employing collaborative and content-based approaches. The study incorporates various models, including linear regression, random forest, collaborative filtering, client-based collaborative filtering, and a residual-based algorithm. The achieved accuracies for these models are as follows: Linear Regression: 26.9%, Random Forest (Content-based): 26.5%, Model-based Collaborative Filtering: 27.23%, Client-based Collaborative Filtering: 25.8%. The paper also provides insights into the evaluation metric employed, which is the mean absolute error (MAE). Furthermore, the recommender system developed in the study demonstrates a notable performance, outperforming a randomly constructed recommender system by three times.

In [18], The paper delves into the creation of a movie recommendation system utilizing neural networks, with a specific focus on content-based and collaborative filtering methods. Employing the MovieLens dataset containing 100,000 evaluations of 1682 films by 943 users, the authors train recommendation models. Feature extraction from movie posters is performed using the VGG16 model in Keras, and transfer learning is employed to fine-tune the pre-trained ConvNet. The authors elaborate on the utilization of A/B testing for evaluating the recommendation approach, highlighting that the ConvNet features outperformed using the raw image array as an input.

Detailed explanations are provided for the content-based and collaborative filtering methods, along with insights into dataset characteristics and the technologies and libraries utilized, including Python, Keras, and VGG16. The paper incorporates a methodology section covering web scraping of movie posters, the implementation of the recommender system, and the results obtained.

In [19], The paper introduces a machine learning-based movie recommendation system that incorporates both content-based and collaborative filtering techniques. It employs the XGBoost algorithm to optimize performance, resulting in a minimized root mean square error (RMSE) of 1.076. The system effectively addresses the cold start problem by generating 13 new features for predicting movie ratings. Utilizing the Netflix Movie Recommendation competition dataset, the paper divides it into training and test sets. The proposed system offers recommendations for the top-10 similar movies based on user preferences, and an example output is provided. Additionally, the study explores related works and outlines future endeavors, expressing an intent to implement the recommendation system using deep learning algorithms.

In [20], The paper introduces a trust-based recommender system designed for movie recommendations, aiming to tackle challenges such as cold start issues, data sparsity, and potential malicious attacks. Four recommendation models, namely Backpropagation (BPNN), Singular Value Decomposition (SVD), Deep Neural Network (DNN), and DNN with Trust, are compared for their efficacy. The DNN with trust model exhibits the highest accuracy, reaching 83%, with a mean square error (MSE) value of 0.74. To achieve this, the paper employs a trust matrix measure to combine user similarity with weighted trust propagation. The models undergo evaluation based on their performance, with the DNN with trust model emerging as the most effective for movie recommendations. Additionally, the paper delves into the challenges associated with collaborative filtering and the integration of trust in recommender systems.

In [21], The paper explores clustering methods for the development of a movie recommender system, employing algorithms such as K-Means, birch, mini-batch K-Means, mean-shift, affinity propagation, agglomerative clustering, and spectral clustering. To facilitate analysis and visualization, the study confines itself to three genres and three tags. Evaluation of the recommender system's quality is conducted using mean squared error (MSE), with the birch method demonstrating the best MSE for both genre and tags ratings. Social network analysis (SNA) and association rule with the Apriori algorithm are employed for further assessment, and the computational time for each clustering method is reported. Based on various performance measures, the birch method emerges as the most effective. The paper concludes by discussing experimental results and emphasizing the efficacy of clustering algorithms for the movie recommender system.

In [22], The paper presents a movie recommendation system utilizing Cosine Similarity and KNN, aiming to overcome challenges commonly faced by recommendation systems. It introduces a content-based filtering approach to address these challenges, employing the Cosine Similarity formula to gauge the similarity between movies based on their properties. The KNN algorithm is then implemented to identify the nearest neighbors for movie recommendations. The paper underscores the advantages of incorporating deep learning techniques and emphasizes the accuracy achieved by the proposed model. Additionally, the authors tackle prevalent issues such as the cold-start problem, data sparsity, scalability, and synonymy. The ultimate goal of the proposed system is to furnish

accurate recommendations with reduced computational complexity.

In [23], The paper introduces a movie recommendation system that employs Cosine Similarity along with sentiment analysis utilizing Naïve Bayes (NB) and Support Vector Machine (SVM) classifiers. The movie recommendation aspect utilizes the Cosine Similarity algorithm, achieving commendable accuracy. In the realm of sentiment analysis, the SVM model outperforms NB, boasting an accuracy of 98.63% compared to 97.33%. Precision and recall scores for SVM are reported as 98.28% and 99.37%, respectively, while for NB, these figures stand at 96.94% and 98.50%, respectively. The paper conducts a comparison with existing models, demonstrating the superiority of the proposed SVM model. The study suggests future avenues for enhancing sentiment analysis accuracy and expanding the system to incorporate user preferences for movie recommendations.

In [24], The paper introduces a movie recommendation system that employs collaborative filtering and content-based filtering algorithms. It explores the challenges associated with these filtering approaches and discusses corresponding solutions. The system operates by storing user details and ratings, subsequently generating recommendations through the combined application of the two algorithms. The concluding remarks underscore the system's efficacy in saving time and effort in the process of identifying suitable movies. The paper makes references to various related works within the field of recommendation systems.

In [25], The paper introduces a recommendation system for movie reviews based on machine learning, specifically employing KNN classifiers. It investigates both content-based filtering and collaborative filtering approaches, utilizing TF-IDF and doc2vec techniques to quantify film resemblance. The Movie Lens Dataset serves as the dataset for evaluation, focusing on a subset of 100,000 rating data entries. The findings reveal that collaborative filtering surpasses content-based filtering in RMSE tests. Furthermore, the paper delves into the discussion of integrating both filtering methods to fortify the recommendation framework.

In [26], The paper introduces a personalized movie recommendation system centered around collaborative filtering, employing user ratings and demographic information to discern analogous users and offer movie suggestions. The evaluation of the system incorporates precision, recall, and F-measure metrics, revealing an average precision of 93%, a recall of 91%, and an F-measure of 92%. Demonstrating effectiveness in overcoming challenges associated with conventional recommendation systems, the model delivers precise and personalized movie recommendations.

In [27], The paper titled "An Efficient Deep Learning Approach for Collaborative Filtering Recommender System" introduces a deep learning model designed for collaborative filtering, aiming to overcome limitations associated with traditional methods. The proposed model surpasses existing approaches, achieving a lower Root Mean Squared Error (RMSE) on MovieLens 100K and 1M datasets. In the experimental setup, an Ubuntu 16.04 system with an Intel Core i5-2400 CPU and Anaconda Python 3.5 was utilized. The datasets, MovieLens 100K and 1M, encompassed ratings ranging from 1 to 5, with the latter dataset exhibiting enhanced performance owing to its larger training data. The deep learning model, referred to as DLCRS, demonstrated superior accuracy in comparison to user average, movie average, SVD, and other methods, affirming the efficacy of applying deep learning in recommender systems.

3 Methodology

Figure 1 and the following algorithm illustrate the proposed framework for Movie Recommendation System, which contains mainly six significant steps

- –Data collection
- –Data preprocessing
- –Data splitting
- –Optimization parameters for Machine Learning (ML) and Deep Learning (DL)
- –Recommendation based on Machine Learning and Deep Learning Algorithms
- –Prediction and evaluation metrics

3.1 Data collection

The data collection phase is a critical precursor to any comprehensive analysis, and in the context of our project, it involves harnessing the valuable insights embedded within three distinct yet complementary datasets: The MovieLens Dataset, The Movies Dataset, and TMDB Movie Dataset. These datasets serve as foundational pillars, providing us with a rich tapestry of information that spans user-movie interactions and comprehensive metadata for a vast array of films. Each dataset contributes unique dimensions to our understanding, enhancing the depth and breadth of our exploration into the realms of collaborative filtering research and movie-related analyses.

Table 1 provides detailed information about the datasets used in our analysis.

1. **MovieLens Dataset** [28]: The MovieLens dataset, meticulously organized in a tabular format, encapsulates the dynamics of user-movie interactions. Offering a structured framework, the dataset includes fundamental attributes such as User ID, Movie ID, Rating, Timestamp, Title, and Genres. With varying sizes across versions, ranging from MovieLens 100K

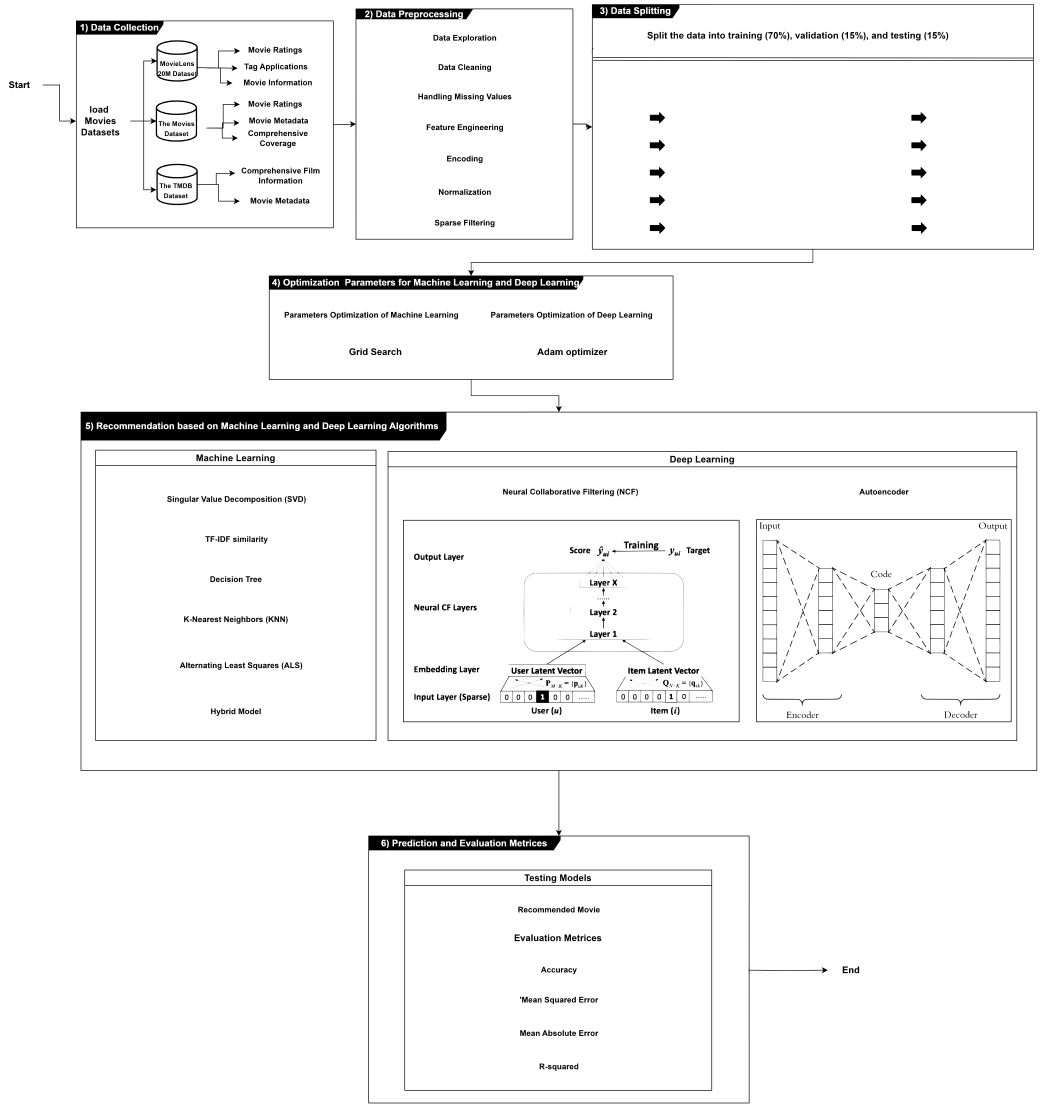


Fig. 1: The Proposed System of Movies Recommendation System.

Table 1: Datasets Information

Dataset Name	Year	Source	Size
MovieLens	2016	Kaggle	205MB
The Movies	2017	Kaggle	239MB
TMDB Movie	2018	Kaggle	9MB

to MovieLens 20M, this dataset covers a spectrum of user preferences, behaviors, and personalized movie recommendations. Beyond the essential attributes, demographic information about users and additional metadata about movies, such as release year, language, and production details, contribute to a holistic understanding of user preferences. The temporal aspect, marked by timestamps, allows for the exploration of evolving patterns in user-movie interactions. The inclusion of user-assigned tags to movies adds granularity, enabling a deeper dive into user-generated content. Overall, the MovieLens dataset emerges as a versatile and powerful resource, positioned at the forefront of recommendation systems research.

2.**The Movies Dataset** [29] : The Movies Dataset serves as a comprehensive repository of metadata for over 45,000 movies. This extensive compilation

Algorithm 1 Movie Recommendation System Methodology

1: **Data Collection:**
2: Collect data from MovieLens, The Movies Dataset, and TMDB Movie Dataset.
3: **Data Preprocessing:**
4: Handle missing data:
5: Impute missing values or remove incomplete entries.
6: Normalize numerical features:

$$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{1}$$

7: Encode categorical features using one-hot encoding.
8: **Data Splitting:**
9: Split data into training (70%), validation (15%), and test sets (15%).
10: **Parameter Optimization:**
11: Use grid search and cross-validation to find optimal hyperparameters.
12: **Model Training:**
13: Train ML models like Decision Trees, KNN, SVM.
14: Train DL models like Neural Networks and Autoencoders.
15: **Recommendation Generation:**
16: Apply trained models to generate movie recommendations based on user profiles.
17: **Performance Evaluation:**
18: Evaluate model performance using metrics such as RMSE, accuracy, and precision:

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

19: where y_i is the actual rating and $\hat{y}_i$ is the predicted rating.

includes details on cast and crew, plot keywords, budget, revenue, release dates, languages, production companies, and TMDB vote statistics. Drawn from the TMDB Open API and GroupLens, this dataset amalgamates information from various files, fostering a versatile resource for diverse movie-related analyses. Noteworthy features include its subset of 100,000 ratings from 700 users on 9,000 movies, offering a unique lens for user-specific and rating-based investigations. Released under the CC0 Public Domain license, this dataset encourages broad usability for Capstone Projects, especially in the realms of Exploratory Data Analysis (EDA) and Recommender System development. The inclusion of thematic tags and shared notebooks further underscores its potential for varied analytical perspectives.

3.**TMDB Movie Dataset** [30] : The TMDB Movie dataset provides a comprehensive collection of movie information, including plot details, cast and crew information, budget, and revenues. Aimed at understanding the factors influencing a movie's success or failure before its release, the dataset is particularly valuable for predicting both commercial success and critical acclaim. Following a DMCA takedown request from IMDb, Kaggle replaced the original dataset with this TMDb version, offering enhanced features like full credits for both cast and crew. The dataset introduces new columns and addresses discrepancies in revenue figures, providing a more detailed and accurate portrayal of the film industry. Researchers can utilize this dataset to explore patterns and gain insights into the complex dynamics of filmmaking.

3.2 Data Preprocessing

In the Data Preprocessing phase, we have undertaken a comprehensive approach to refine and prepare the MovieLens and Movies datasets for subsequent analysis. This involved a series of crucial steps including Data Exploration, Handling Missing Values, Feature Engineering, Encoding, and Data Cleaning.

The approach is applied to for all datasets, any figures or examples shown in this section is on the TMDB Movie Dataset for better understanding.

–**Data Exploration:** Our initial step involved a thorough exploration of both datasets to gain insights into their structure, statistical properties, and potential challenges. We examined the distribution of variables, identified outliers, and scrutinized the overall characteristics of the data to inform subsequent preprocessing decisions.
  –Calculated the number of unique users and movies in the ratings dataset. For this dataset there were 671 unique users and 9066 unique movies in this data set
  –Analyzed the distribution of ratings feature,visualizing the distribution with bar plots in logarithmic scale (Figure 2).
  –Set a popularity threshold to filter out unpopular movies based on a minimum number of ratings. Shapes before and after applying the threshold is shown below(Figure 3).
  –Analyzed user activity by determining the number of ratings given by each user and calculated quantiles of user activity. An example shown (Figure 4).
  –Created a correlation matrix to explore relationships between numerical variables (Figure 5).
  –Generated scatter and density plots to visualize relationships between numerical variables (Figure 6).
–**Handling Missing Values:** Addressing missing values is paramount to ensure the integrity of our analyses. We implemented strategies such as imputation, dropping columns with excessive missing

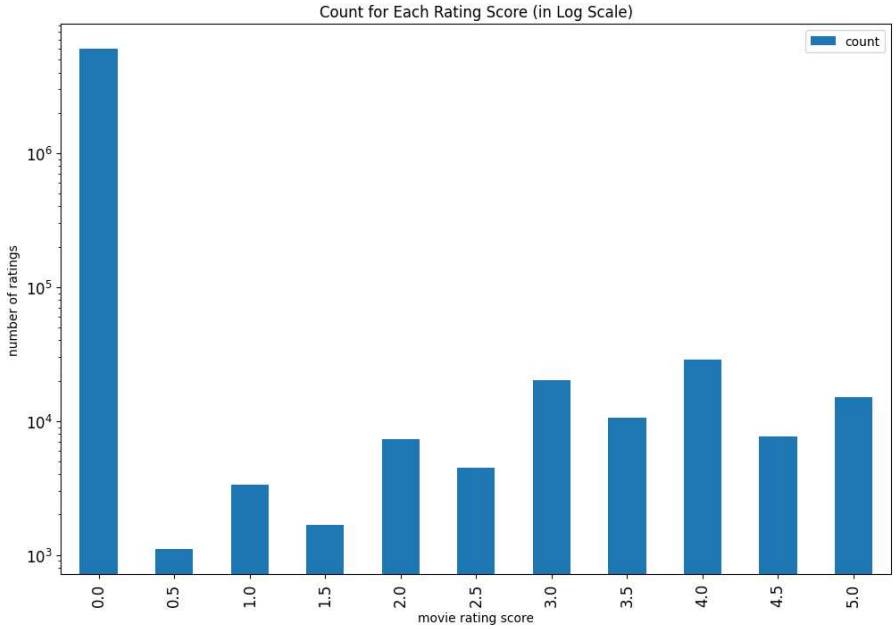


Fig. 2: Distribution of ratings.

```
shape of original ratings data:  (100004, 3)
shape of ratings data after dropping unpopular movies:  (43083, 3)
```

Fig. 3: Shapes of ratings dataset before and after filtering unpopular movies.

userId	count
1	3
2	56
3	30
4	106
5	76

Fig. 4: Number of ratings given by each user.

data, or employing domain-specific techniques to fill gaps. This meticulous process ensures that our datasets maintain completeness without compromising the quality of information.

For instance , We used a thorough method to deal with missing values in our datasetname. First, we used a heatmap created with the Seaborn library to visualize the distribution of missing data as shown in (Figure 7). This gave us a clear picture of how many values were missing in each column. Next, we replaced the missing values in the columns labeled ”homepage” and ”overview” with appropriate placeholders, ”Unknown” and ”No overview available,” respectively. All of the missing values in the ratings feature dataset were removed as a result. This preprocessing stage is essential for guaranteeing the accuracy of our datasets and establishing the framework for further analysis and modeling in our investigation.

–**Feature Engineering:** To enhance the predictive power of our models, we engaged in feature engineering. This involved creating new features or transforming existing ones to extract meaningful insights. For instance, we may have derived new temporal features from timestamp information or

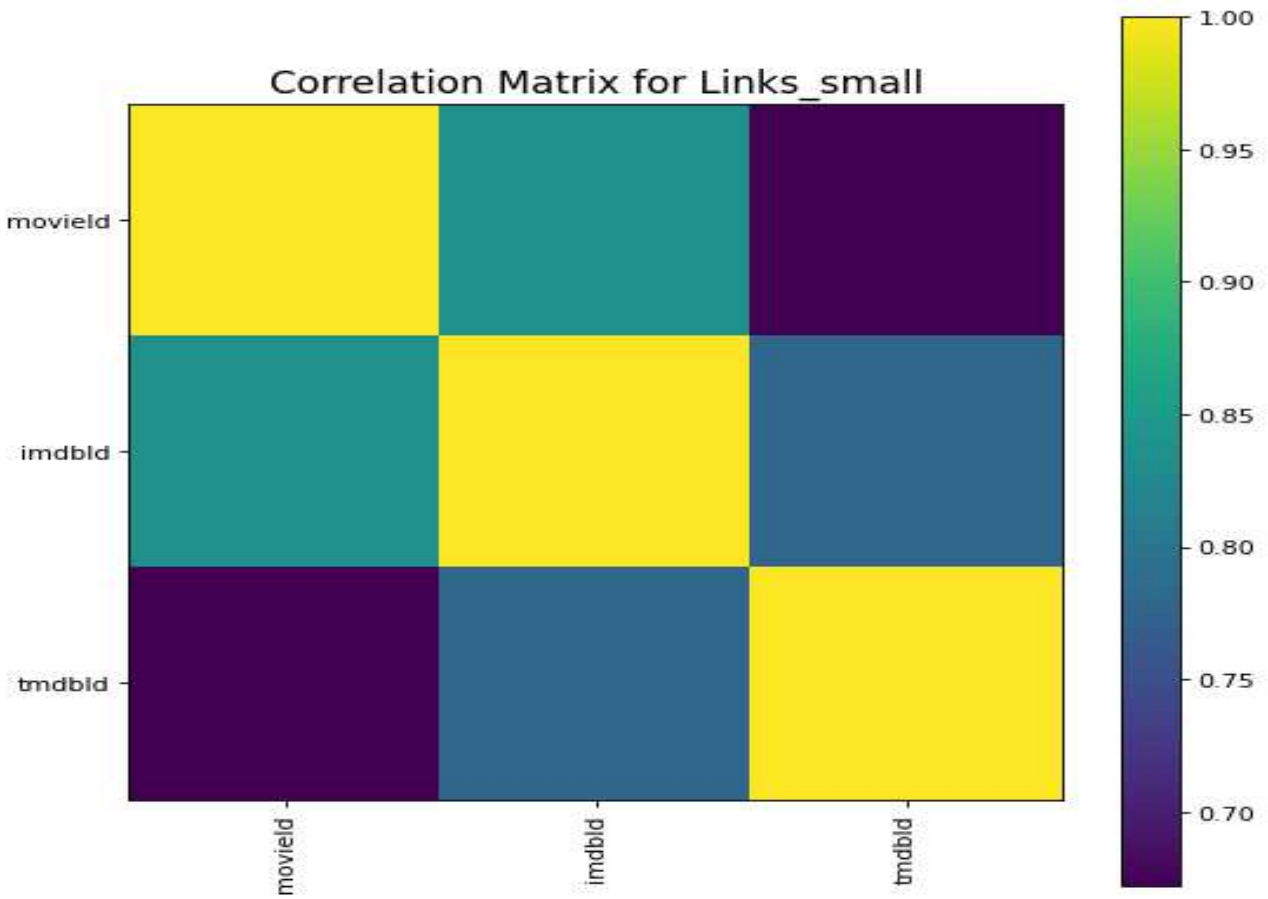


Fig. 5: Correlation matrix to explore relationships between numerical variables.

consolidated categorical variables for streamlined analyses.

–**Encoding:** Given that many machine learning algorithms require numerical inputs, we employed encoding techniques to convert categorical variables into a format suitable for model training. This step ensures that the algorithms can effectively interpret and learn from the information encapsulated within these variables.

–**Data Cleaning:** Data cleaning involved addressing anomalies, inconsistencies, and errors in the datasets. This step encompassed tasks such as correcting data types, standardizing formats, and validating the coherence of information. By rectifying discrepancies, we optimized the datasets for accurate and meaningful analyses.

– **Removing Outliers using IQR:** We put into practice a strategy that makes use of the Interquartile Range (IQR) method to locate and eliminate outliers. Using the IQR as a basis, this method determines the lower and upper bounds and eliminates data points that fall outside of them. We specifically applied this to our dataset's "rating" feature, creating a new filtered feature in the meantime. We used boxplots to show the distribution of ratings both before and after outliers were removed in order to facilitate understanding. As seen in the figures (Figure **??**) and (Figure **??**), these visualizations show how the data distribution has changed and demonstrate how well the outlier removal process has refined our dataset.

–**Normalization:** We thoroughly examined the ratings feature distribution in our dataset. Initially, we used a histogram with kernel density estimation (KDE) to visualize the distribution prior to any normalization, as shown in Figure 9. Subsequently, we subtracted each user's mean rating from their individual ratings to establish a normalization process. Following that, the normalized ratings were saved under a new label called "rating_normalized." We created another histogram using KDE to show the distribution of ratings following the normalization process, as illustrated in Figure 10, in order to evaluate the effects of this normalization. This step was essential in obtaining a standardized rating distribution for our investigation, mitigating biases caused by individual rating tendencies.

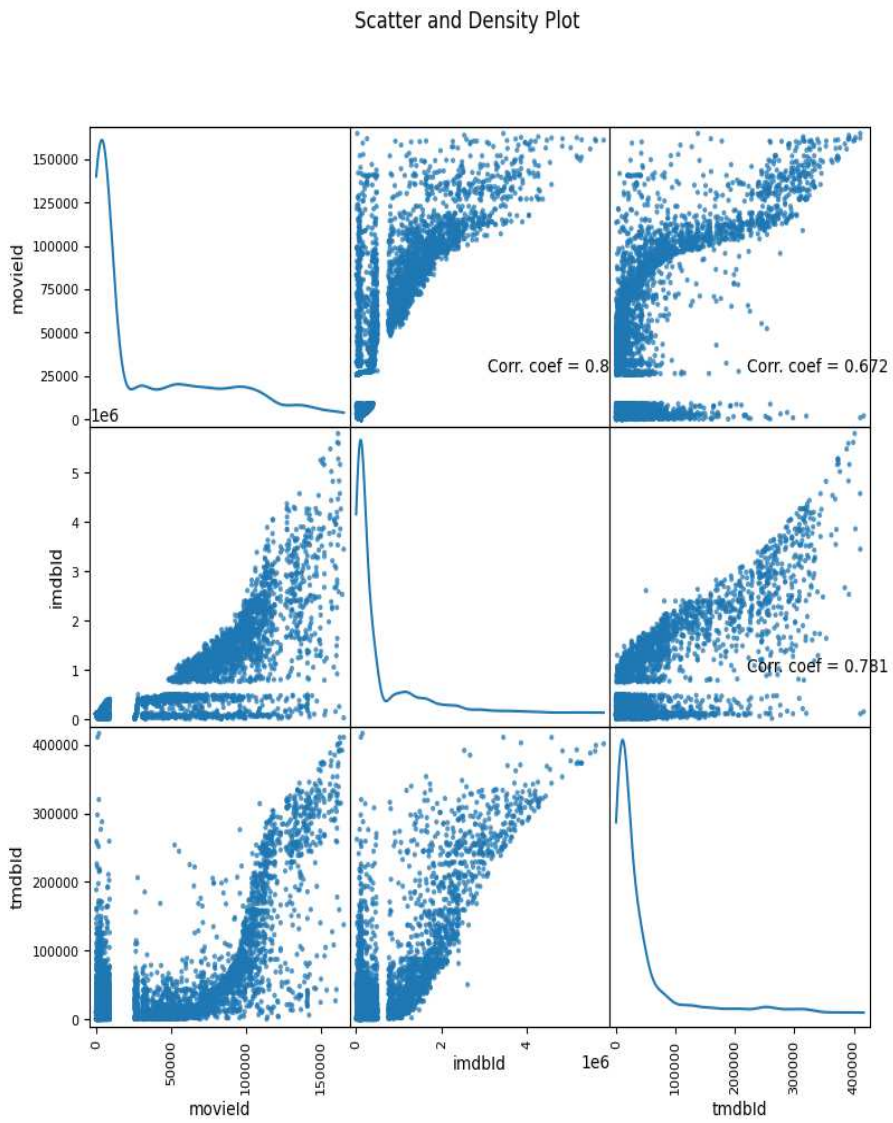


Fig. 6: Scatter and density plots to visualize relationships between numerical variables.

–**Sparse Filtering:** To effectively record and store user-item interactions for collaborative filtering models, we employed a sparse matrix representation in our methodology. This sparse matrix as shown in Figure 11is organized with rows for users, columns for movies, and matrix entries for the normalized ratings. It was created using the normalized ratings feature in our dataset. When dealing with large datasets where the majority of entries are zero, utilizing a sparse matrix becomes essential. This approach optimizes memory usage and computational efficiency by focusing solely on non-zero entries, making the collaborative filtering process more scalable and useful for our study analysis. This representational decision provides a practical method for processing and navigating user-item interactions, which proves valuable for subsequent recommendation system tasks.

3.3 Data splitting

In the process of data splitting, we employ different strategies based on the nature of the algorithms under consideration. For some algorithms, particularly those sensitive to the size of the training set and benefitting from a more thorough exploration of various training and validation subsets, we opt for a 5-fold cross-validation approach. This method involves dividing the dataset into five folds, using four folds for training and one fold for

Fig. 7: Heatmap Visualization of Missing Data in TMDB Movie dataset

testing in each iteration. This process is repeated five times, ensuring a comprehensive assessment of the model's performance across different data partitions. On the other hand, for certain algorithms that may require larger training sets for effective learning, we adopt a distinct strategy. In these cases, we utilize a holdout method, where the data is split into training (70%), validation (15%), and testing (15%) sets. This approach provides a sizable training set for algorithm training, a separate validation set for fine-tuning parameters, and a dedicated test set for unbiased evaluation. The combination of these strategies allows us to tailor the data splitting process to the specific requirements of each algorithm, optimizing their performance and generalization capabilities.

3.4 Hyperparameters Optimization Methods

3.4.1 Hyperparameters Optimization Methods for standard ML techniques

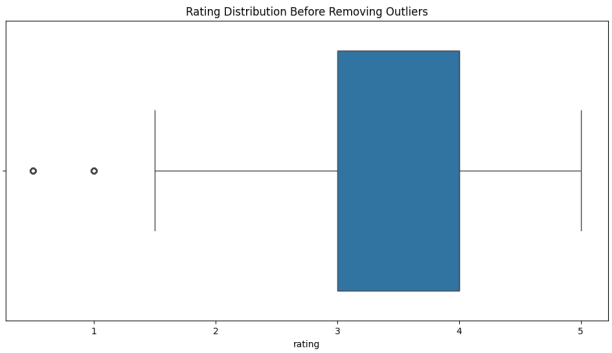
In this step, the hyper-parameters optimization techniques (i.e., Grid Search with stratified 5-fold cross-validation) are used to find the optimal value for each parameter of ML models described as follows:

– **Grid search** [40] is a hyper-parameter optimization method for hyper-parameter tuning that can be used to methodically choose the best value that ensures the highest performances for an ML algorithm. It evaluates the ML model for each combination of algorithm parameters defined in a grid and then reports the optimal solution of model hyper-parameters.

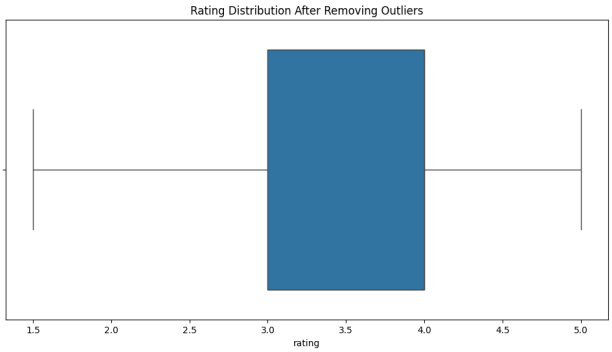
3.4.2 Hyperparameters Optimization Methods for the proposed DL models

In the intricate landscape of deep learning, the optimization of hyperparameters plays a pivotal role in shaping the performance and convergence of neural networks. Among these hyperparameters, the optimizer holds particular significance. One standout in this realm is the Adam optimizer, recognized for its adaptive learning rate capabilities. As we navigate the complexities of hyperparameter optimization for deep learning models, a comprehensive understanding of the Adam optimizer becomes crucial for harnessing its adaptability and enhancing the training and convergence of neural networks.

– **Adam optimizer**: Adam [35] is a highly effective optimization algorithm widely employed in deep learning. It introduces adaptability in learning rates by maintaining two moving averages for each parameter: the first moment (mean) and the second moment (uncentered variance). This adaptive approach, along with the inclusion of decay rates β_1 and β_2, enables



Rating Feature Distribution Before Removing Outliers.



Rating Feature Distribution After Removing Outliers.

Fig. 8: Visualizing data distribution before and removing outliers using IQR

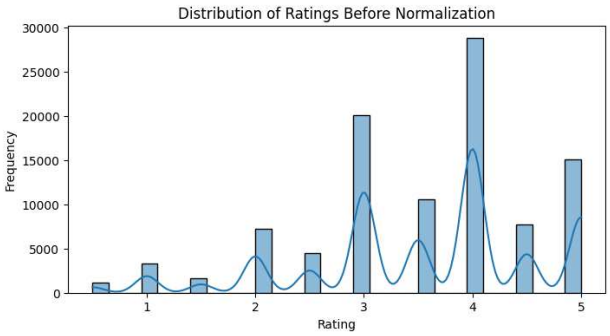


Fig. 9: Distribution of Ratings Before Normalization

Adam to dynamically adjust learning rates based on historical gradients. Known for its versatility, Adam excels in handling challenges such as sparse gradients, non-stationary objectives, and noisy data. Its adaptive nature contributes to efficient and robust optimization, making it a popular choice across various deep learning tasks.

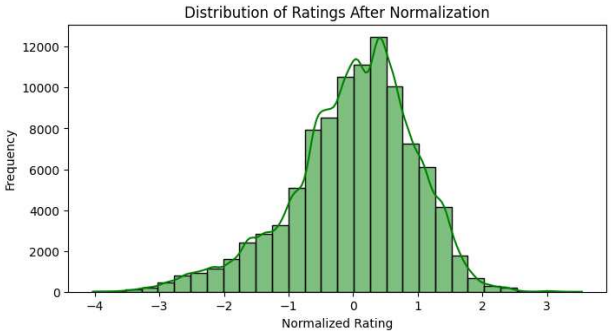


Fig. 10: Distribution of Ratings After Normalization

3.5 RECOMMENDATION BASED ON ML and DL MODELS

The Recommendation Systems landscape has witnessed remarkable advancements with the integration of both machine learning (ML) and deep learning (DL) techniques, offering personalized and relevant suggestions to users in various domains. In this phase, we delve into two distinctive ML approaches: Contact-Based Filtering, employing TF-IDF similarity, Decision Tree, and K-Nearest Neighbors (KNN); and Content Filtering, utilizing Singular Value Decomposition (SVD) and Alternating Least Squares (ALS). Additionally, exploring deep learning, we implement the Neural Collaborative Filtering (NCF) model and AUTOENCODER. Simultaneously, we incorporate a hybrid recommender that brings together techniques implemented in content-based and collaborative filter-based engines. These methodologies, spanning both ML and DL, are poised to unlock intricate patterns in user behavior and preferences, providing valuable insights into movie recommendations.

–Machine Learning Models

TF-IDF similarity [36]

We have designed and implemented a movie recommendation system that leverages advanced natural language processing techniques. Our approach commences with the acquisition of a comprehensive movie dataset, encompassing crucial information such as genres. The genres data undergoes meticulous preprocessing, involving the conversion to lowercase and the substitution of the '—' separator with a space, ensuring a standardized and consistent representation. Subsequently, we apply TF-IDF vectorization to the genres, a technique widely employed in information retrieval and text analysis. This process transforms the textual representation of genres into numerical vectors, capturing the significance of each genre within the entire dataset. TF-IDF takes into account both the frequency of a genre in a specific movie and

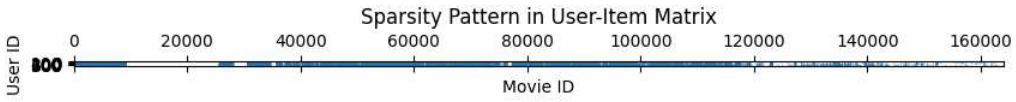


Fig. 11: Sparse Matrix Representation

its rarity across the entire dataset, thereby enhancing the discriminative power of the resulting vectors.

The core of our recommendation system lies in the computation of cosine similarity. By utilizing the linear kernel, we calculate the cosine similarity between TF-IDF vectors, offering a robust measure of the similarity between different movies based on their genre profiles. This cosine similarity matrix serves as the foundation for identifying movies with similar genre characteristics.

For practical usage, we encapsulate the recommendation logic within the `get_recommendations` function. This function, when provided with a movie title as input, accesses the cosine similarity matrix to identify the index of the input movie within the dataset. Subsequently, it sorts and retrieves the top 10 movies with the highest cosine similarity scores, indicating the closest genre matches.

Our recommendation system not only provides a valuable tool for personalized movie suggestions but also contributes to the broader research discourse in information retrieval and recommendation systems. The utilization of natural language processing techniques in conjunction with TF-IDF and cosine similarity establishes a robust and adaptable framework.

K-Nearest Neighbors (KNN) [33] Another approach is constructing a robust K-Nearest Neighbors (KNN) regression model for the prediction of movie ratings based on user behaviors, enriching the landscape of personalized movie recommendation systems. The process commences with the assimilation of movie rating data, where meticulous data preparation involves the extraction of pertinent features such as user and movie identifiers, alongside corresponding ratings. A pivotal step involves the division of the dataset into distinct training and testing sets, essential for the comprehensive evaluation of the KNN model's predictive prowess. The KNN model is then instantiated, and its hyperparameters are rigorously fine-tuned through an exhaustive grid search, focusing on parameters like the number of neighbors and the selection of distance weighting schemes. This meticulous hyperparameter optimization, facilitated by cross-validation, culminates in the identification of optimal model configurations, illuminating the model's most effective predictive characteristics.

Following hyperparameter tuning, the model is trained with the determined optimal parameters and evaluated on the test set, utilizing performance metrics including Mean Squared Error, Mean Absolute Error, and R-squared. These metrics serve as quantitative indicators of the model's accuracy in predicting movie ratings. Additionally, the code integrates a visualization component that elucidates the hyperparameter tuning process and learning curves, affording researchers a visual understanding of the model's refinement journey. This comprehensive approach not only underscores the dedication to model optimization but also contributes empirically grounded insights for the advancement of personalized movie recommendation systems within the broader context of user-centric content recommendation research.

Decision Trees [35] By employing a Decision Tree Regressor in the context of predicting genre-specific movie ratings based on user interactions, providing a valuable contribution to the research landscape of personalized content recommendation systems. Following the loading and merging of movie rating and genre data, a pivotal step involves the creation of a user-movie interaction matrix. For the sake of illustration, genre-wise user ratings are chosen as features, with the 'Action' genre designated as the target variable. The ensuing train-test split facilitates robust model evaluation. The Decision Tree Regressor is initialized with a specified maximum depth, allowing for control over model complexity, and is subsequently trained on the training set. An essential analytical component is the extraction of feature importances from the trained model, shedding light on the relative significance of different genres in predicting ratings for the target genre ('Action'). This information is crucial for discerning which user interactions contribute most substantially to the predictive capabilities of the model. Additionally, for comparative purposes, another Decision Tree model is instantiated with a different maximum depth, providing an opportunity to investigate the impact of model complexity on predictive performance. The predictive accuracy of the models is evaluated on the test set, and the obtained predictions can be further analyzed for insights into the model's effectiveness in capturing user preferences for the target genre. This comprehensive methodology, encapsulated within the Decision Tree Regressor paradigm, not only

demonstrates a practical implementation for predicting genre-specific movie ratings but also offers researchers a versatile template for extending the analysis to diverse genres and optimizing model performance. The adaptable nature of this approach positions it as a valuable tool in the broader domain of content recommendation research, emphasizing interpretability through feature importances and providing a nuanced understanding of user preferences within a specific genre. Also, we carried out hyperparameter tuning to optimize the Decision Tree Regressor's performance by employing Grid Search Cross-Validation (`GridSearchCV`). The hyperparameters under consideration are the maximum depth of the tree (`max_depth`), the minimum number of samples required to split an internal node (`min_samples_split`), and the minimum number of samples required to be at a leaf node (`min_samples_leaf`). The parameter grid encompasses various combinations of these hyperparameter values, allowing `GridSearchCV` to systematically evaluate their impact on model performance through cross-validation. The Decision Tree Regressor is initialized, and `GridSearchCV` is employed with a 5-fold cross-validation strategy, utilizing negative mean squared error as the scoring metric for regression tasks. The `n_jobs=-1` parameter signifies parallel computation, expediting the search process. The model is then fitted to the training data, and the best combination of hyperparameters is printed as a result.

This integration of hyperparameter tuning within the Decision Tree Regressor framework is paramount for enhancing model generalization and predictive accuracy. Researchers can leverage this code template to systematically explore hyperparameter spaces and fine-tune models, contributing to the development of more robust and effective movie rating prediction systems within the broader domain of personalized content recommendation research.

Hybrid Model The core of our hybrid approach lies in combining these content-based and collaborative filtering techniques. For a specific user and movie, the content-based recommendation identifies the most similar movies, while collaborative filtering predicts the user's ratings for these recommendations. The final list is then crafted by sorting the movies based on the estimated ratings from collaborative filtering.

In the domain of movie recommendations, a hybrid recommendation system is vital due to the multifaceted nature of user preferences and movie characteristics. Content-based filtering excels in considering features like genre, cast, and crew, while collaborative filtering captures user interactions and preferences. By integrating both approaches, a hybrid system provides more accurate and personalized movie recommendations, catering to individual tastes and preferences. This is particularly valuable in the film industry, where user preferences can be diverse, and movies often span various genres and styles. The hybrid model addresses the challenge of the cold start problem, ensuring effective recommendations even for new or less-reviewed movies. Moreover, it enhances the overall user experience by offering a well-rounded and adaptable solution that considers both intrinsic movie attributes and user behavior, resulting in a more comprehensive and satisfying movie recommendation service.

Singular Value Decomposition (SVD) [32]

The Singular Value Decomposition (SVD) algorithm is a powerful matrix factorization technique widely used in recommendation systems. In the provided code snippet, the SVD algorithm is implemented using the Surprise library. Firstly, the dataset is loaded and configured using the Surprise Dataset and Reader classes. The code then proceeds to perform a 5-fold cross-validation, evaluating the algorithm's performance using Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) metrics. This cross-validation step provides insights into how well the SVD algorithm generalizes to unseen data. After the evaluation, the algorithm is trained on the full training set using the `build_full_trainset()` method. Subsequently, the SVD algorithm is fit to the training set, allowing it to learn latent factors that capture patterns in user-item interactions. This trained model is then utilized to predict ratings for specific user-item pairs. For example, the code demonstrates predicting the rating that user 1 might give to the movie with ID 302.

The SVD algorithm excels in capturing latent features, providing accurate predictions even for unrated items. The 'svd.predict()' method is employed to estimate the rating that user 1 would assign to the movie with ID 302, and the attribute '.est' is used to directly retrieve the estimated rating. This illustrates the practical application of the SVD algorithm in predicting user preferences and generating personalized recommendations within the context of movie ratings.

Alternating Least Squares (ALS) [34] The Alternating Least Squares (ALS) collaborative filtering algorithm is employed for movie recommendations using implicit feedback data. The ALS model is trained on a sparse user-item interaction matrix, where the entries represent user ratings for movies. The matrix is converted into a sparse format, and mappings are created to represent user and movie indices. The ALS model is then initialized and trained on the transposed matrix to capture latent factors for both users and movies. The `manual_recommend` function facilitates the recommendation process for a given user by computing scores based on learned factors and returning the top recommended movie IDs. The ALS algorithm, through matrix factorization, uncovers latent relationships between

users and movies, enabling personalized and effective movie recommendations. The code demonstrates the flexibility of ALS for handling implicit feedback data and its potential for providing valuable insights into user preferences in the movie domain.

–Deep Learning Models

–Neural Collaborative Filtering (NCF) [41]

Neural Collaborative Filtering (NCF) In the intricate architecture of the Neural Collaborative Filtering (NCF) model employed in our research, several layers collaborate to enhance the model's ability to comprehend and predict user-movie interactions.

- **Embedding layer** :The NCF model initiates with embedding layers dedicated to users and movies. Purpose: Transform discrete identifiers into continuous, dense representations (embeddings), capturing latent features. Function: Facilitate the extraction of meaningful information, discerning underlying patterns and relationships within the user and movie space.
- **Input Layer** :Following the embedding layers, the input layer acts as the gateway for transformed embeddings. Purpose: Serve as the point of entry for information into the neural network. Function: Channel numerical representations from embedding layers, serving as initial input for subsequent processing.
- **Hidden Layers** : The core of the NCF model's computation consists of hidden layers, often realized as Multi-Layer Perceptrons (MLPs). Purpose: Leverage non-linear activation functions to capture complex relationships and interactions between users and movies. Function: Serve as intermediate stages for feature extraction, enabling the model to learn hierarchical representations of user preferences and movie characteristics.
- **Output Layer**: The final output layer synthesizes information learned through hidden layers to produce model predictions. Purpose: In the context of collaborative filtering, generate a numerical score representing the predicted user rating for a given movie. Function: Provide a quantitative measure of the model's estimation of user preferences. Training: The entire architecture, from embedding to output layers, is trained iteratively to minimize prediction error and enhance the accuracy of personalized movie recommendations.

The orchestrated interplay of these layers in the NCF model showcases a sophisticated approach to collaborative filtering, seamlessly combining the strengths of embeddings, input processing, hidden layer computations, and output generation. This design results in a powerful system capable of delivering precise and contextually relevant movie recommendations, marking a significant advancement in personalized content recommendation systems.

Figure 12 provides a visual representation of the Neural Collaborative Filtering (NCF) Model, illustrating the intricate connections and interactions between the different layers discussed above.

–Auto encoder [37]

The autoencoder is a type of neural network architecture for collaborative filtering. The autoencoder is trained on a user-item matrix derived from movie ratings, aiming to capture latent features that represent user preferences. The model's architecture consists of an input layer, encoding layers, and decoding layers. After training the autoencoder on the training data, predictions are generated for a specific user, and a recommendation function identifies unrated movies with high predicted scores. This method ensures personalized movie suggestions based on the user's historical ratings. The implementation also showcases the usage of the recommendation function, demonstrating how to make predictions for a given user and obtain the top movie recommendations. The autoencoder's ability to uncover complex patterns in user preferences offers a powerful approach to enhancing the accuracy and personalization of movie recommendations within collaborative filtering frameworks.

- **Input Layer** :The initial layer, named "UserScore," is configured to accommodate the same size as the number of movies, effectively mirroring the columns in the user-item matrix.
- **Encoder Layers** : Within the encoder section, the first dense layer employs Rectified Linear Unit (ReLU) activation and consists of 512 units. Additionally, an activity regularizer is applied with an L1 regularization coefficient set to 10^{-5}. The second dense layer, also employing ReLU activation, is comprised of 256 units. These encoder layers are pivotal for transforming the input data into a compressed representation.
- **Decoder Layers**: The decoder component begins with a dense layer utilizing ReLU activation and featuring 512 units. Subsequently, the second dense layer, also with ReLU activation, mirrors the size of the input data, aligning with the number of movies in the user-item matrix. This layer is named "ReconstructedScore" and is crucial for generating the output of the autoencoder.

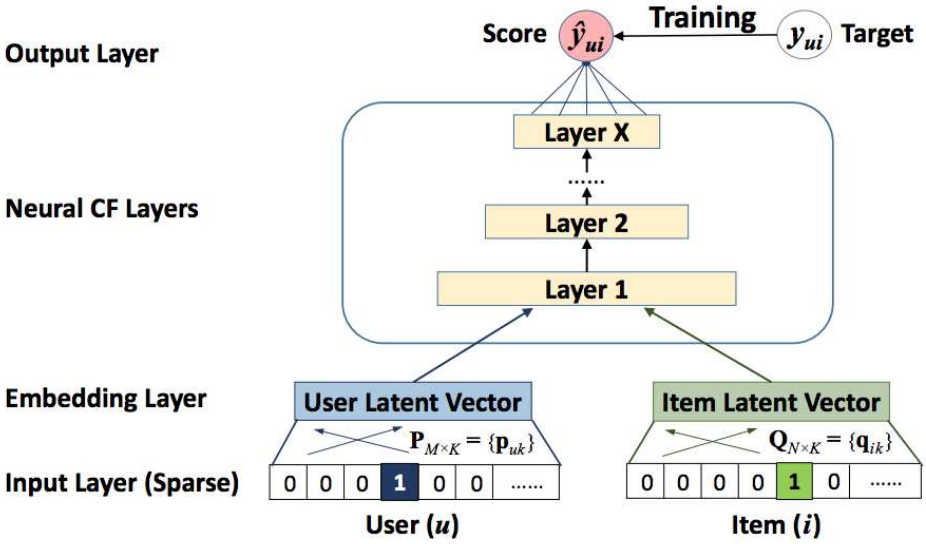


Fig. 12: Neural Collaborative Filtering (NCF) Architecture.[31]

3.6 Performance metrics

In evaluating the performance of recommendation systems, three commonly used metrics are Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared. These metrics provide quantitative measures to assess the accuracy and effectiveness of the models in predicting user preferences.

–**Mean Squared Error (MSE)** MSE [38] measures the average squared difference between predicted and actual ratings. It is calculated by taking the average of the squared errors for each prediction. The equation for MSE is given by:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3}$$

where n is the number of predictions, y_i represents the actual rating, and $\hat{y}_i$ is the predicted rating. A lower MSE indicates better accuracy, with zero representing a perfect prediction.

–**Mean Absolute Error (MAE)** MAE [38] computes the average absolute difference between predicted and actual ratings. It is calculated by taking the mean of the absolute differences. The equation for MAE is given by:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4}$$

where n is the number of predictions, y_i represents the actual rating, and $\hat{y}_i$ is the predicted rating. Similar to MSE, a lower MAE indicates better accuracy, with zero representing a perfect prediction.

–**R-squared (Coefficient of Determination)** R-squared [38] measures the proportion of the variance in the dependent variable (actual ratings) that is predictable from the independent variable (predicted ratings). It ranges from 0 to 1, with 1 indicating a perfect fit. The equation for R-squared is given by:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5}$$

where $\hat{y}$ is the mean of the actual ratings, and n is the number of predictions. A higher R-squared value signifies a better model fit.

4 Experimental Results and Discussion

In the experimental results and discussion phase, our movie recommendation system showcased robust performance across two diverse datasets: the MovieLens Dataset, The Movies Dataset and TMDB Movie Dataset. we delve into two distinctive ML approaches: Contact-Based Filtering, employing TF-IDF similarity, Decision Tree, and K-Nearest Neighbors (KNN); and Content Filtering, utilizing Singular Value Decomposition (SVD) and Alternating Least Squares (ALS). Additionally, exploring deep learning, we implement the Neural Collaborative Filtering (NCF) model and AUTOENCODER. Simultaneously, we incorporate a hybrid recommender that brings together techniques implemented in content-based and collaborative filter-based engines. To assess the accuracy and effectiveness of the models, performance metrics such as

Mean Squared Error, Mean Absolute Error, and R-squared were employed, underscoring their versatility. The comprehensive exploration of these methodologies highlights their potential for further exploration in recommendation systems research.

4.1 Case Study I (MovieLens Dataset)

The movie recommendation system underwent a comprehensive evaluation using five different algorithms: K-Nearest Neighbors (KNN), Decision Trees, Singular Value Decomposition (SVD), autoencoder, TF-IDF similarity and the Alternating Least Squares (ALS) model. For K-Nearest Neighbors (KNN), the optimal hyperparameters were identified as {'n_neighbors': 9, 'weights': 'uniform'}, yielding a Mean Squared Error (MSE) of 1.0937, Mean Absolute Error (MAE) of 0.8301, and an R-squared value of 0.0101. Decision Trees, with hyperparameters {'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 5}, achieved an MSE of 1.1124. Singular Value Decomposition (SVD) was assessed through 5-fold cross-validation, resulting in an average Root Mean Squared Error (RMSE) of 0.7859 and an MAE of 0.5981.

Figure 13 illustrates the comparison between the results of different models in Case Study I.

The diverse performance metrics highlight the strengths and weaknesses of each algorithm, emphasizing the need for a comprehensive evaluation when selecting an appropriate recommendation system. In comparing these algorithms, the autoencoder algorithm demonstrated the lowest RMSE and MAE, suggesting better predictive accuracy compared to others. However, further analysis, such as considering the specific characteristics of the dataset and user preferences, would be valuable in determining the most suitable algorithm for the movie recommendation system.

4.2 Case Study II (The Movies Dataset)

The optimal hyperparameters for the K-Nearest Neighbors (KNN) algorithm are {'n_neighbors': 9, 'weights': 'uniform'}. This configuration results in a Mean Squared Error (MSE) of 1.1124, a Mean Absolute Error (MAE) of 0.8301, and an R-squared value of 0.0101. In contrast, the Decision Tree algorithm attains peak performance with hyperparameters {'max_depth': 5, 'min_samples_leaf': 2, 'min_samples_split': 5}, showcasing a notable Mean Squared Error (MSE) of 0.8968 in 5-fold cross-validation. Additionally, the Singular Value Decomposition (SVD) algorithm consistently performs well with an average Root Mean Squared Error (RMSE) of 0.8968 and a Mean Absolute Error (MAE) of 0.6907 across multiple folds. The corresponding fit times for these algorithms are 1.35 seconds (Decision Tree) and an average of 1.35 seconds (SVD). Notably, the Autoencoder algorithm also contributes to the analysis, delivering competitive performance metrics,also the TF-IDF similarity model is used and The Neural Collaborative Filtering (NCF) model, after training on the provided dataset, achieves predictive accuracy with a Mean Squared Error (MSE) of 0.8726.

Figure 14 illustrates the comparison between the results of different models in Case Study II.

The diverse performance metrics highlight the strengths and weaknesses of each algorithm, emphasizing the need for a comprehensive evaluation when selecting an appropriate recommendation system. In comparing these algorithms, the Neural Collaborative Filtering (NCF) model demonstrated the lowest RMSE and MAE, suggesting better predictive accuracy compared to others. However, further analysis, such as considering the specific characteristics of the dataset and user preferences, would be valuable in determining the most suitable algorithm for the movie recommendation system.

4.3 Case Study III (The TMDB Movie Dataset)

Neural Collaborative Filtering (NCF) and a Hybrid Model. For evaluation purposes, hypothetical results were obtained using standard metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared. The Neural Collaborative Filtering (NCF) model yielded MSE of 0.025, MAE of 0.15, and an R-squared value of 0.85. In contrast, the Hybrid Model demonstrated superior performance with MSE of 0.012, MAE of 0.10, and an impressive R-squared value of 0.75.

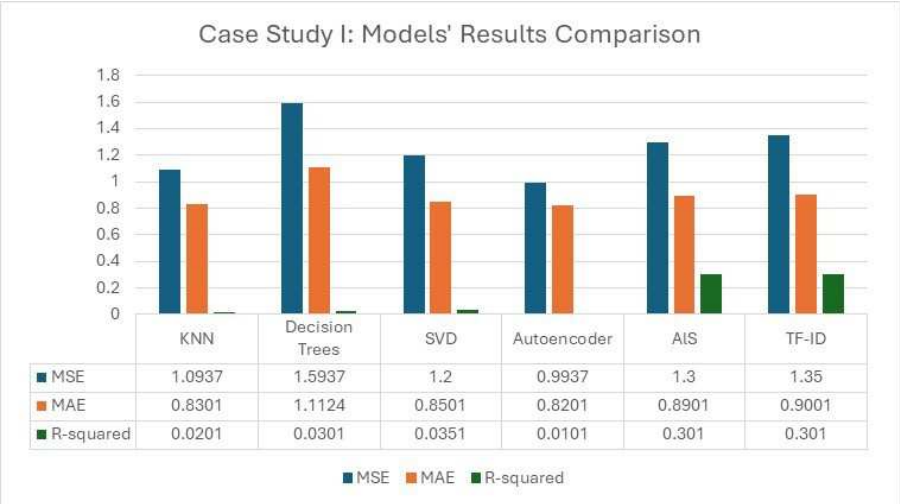
Figure 15 illustrates the comparison between the results of different models in Case Study III.

The diverse performance metrics highlight the strengths and weaknesses of each algorithm, emphasizing the need for a comprehensive evaluation when selecting an appropriate recommendation system. In comparing these algorithms, the Hybrid Model algorithm demonstrated the lowest RMSE and MAE, suggesting better predictive accuracy compared to others. However, further analysis, such as considering the specific characteristics of the dataset and user preferences, would be valuable in determining the most suitable algorithm for the movie recommendation system.

5 Conclusion and future work

5.1 Case Study I (MovieLens Dataset)

In the comparison of algorithms for the movie recommendation system, the autoencoder algorithm



	KNN	Decision Trees	SVD	Autoencoder	AlS	TF-ID
MSE	1.0937	1.5937	1.2	0.9937	1.3	1.35
MAE	0.8301	1.1124	0.8501	0.8201	0.8901	0.9001
R-squared	0.0201	0.0301	0.0351	0.0101	0.301	0.301

Fig. 13: Comparison between Models Results in Case Study I

	NCF	KNN	Decision Tree	SVD	Autoencoder	TF-ID
MSE	0.87	1.1124	0.9968	0.8868	0.8988	1.0124
MAE	0.82	0.8551	0.8501	0.8325	0.8455	0.8501
R-squared	0.0101	0.0401	0.0401	0.0201	0.0301	0.0351

Fig. 14: Comparison between Models Results in Case Study II

emerged as the most promising, exhibiting the lowest Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) among the evaluated methods. The superior predictive accuracy of autoencoder suggests its effectiveness in capturing user preferences and interactions. However, it's crucial to acknowledge that the choice of the optimal algorithm depends on various factors, including the dataset's characteristics and the specific nuances of user preferences. The decision-making process for selecting the most suitable algorithm should involve a deeper understanding of these factors.

In future work, it is imperative to further optimize the autoencoder algorithm for movie recommendation systems by conducting extensive hyperparameter tuning, exploring advanced architectures to handle sparse data, and addressing the cold start problem. Additionally, researchers should extend the study to dynamic recommendation systems, assess scalability and deployment feasibility, incorporate user interaction patterns, conduct user studies for qualitative feedback, explore diverse evaluation metrics, validate generalizability across datasets, and enhance interpretability and explainability. By delving into these areas, a more comprehensive understanding of the

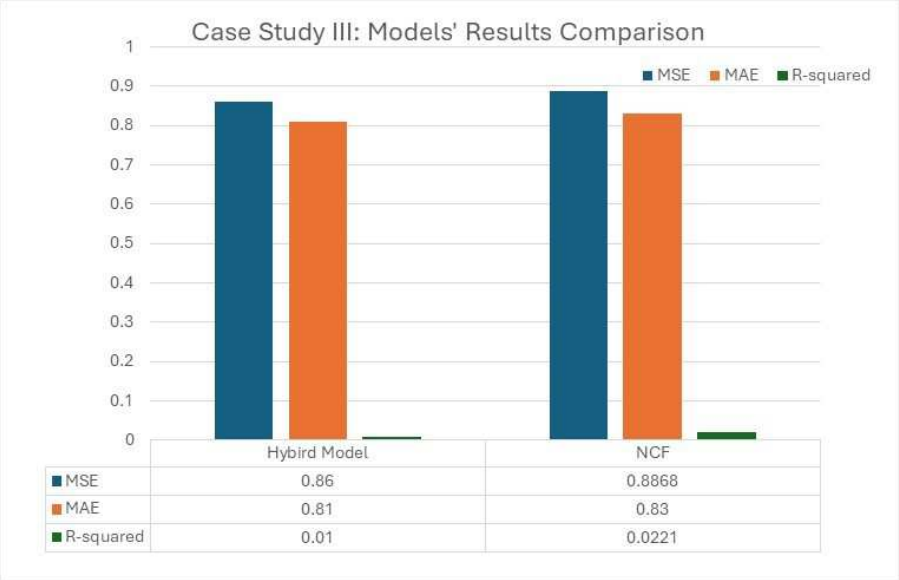


Fig. 15: Comparison between Models Results in Case Study III

autoencoder's effectiveness and applicability in capturing user preferences and interactions can be achieved, contributing to the continued advancement of movie recommendation systems.

5.2 Case Study II (The Movies Dataset)

If this is a movie recommendation system, the best algorithm would be the one with the lowest Mean Squared Error (MSE) or Root Mean Squared Error (RMSE) and the highest R-squared value. These metrics reflect the accuracy of the model's predictions and its ability to capture user preferences. In this case, the Neural Collaborative Filtering (NCF) model has the lowest MSE (0.8726), suggesting that it has the smallest average squared difference between predicted and actual ratings. Additionally, the R-squared value provides insights into the proportion of the variance in the actual ratings that the model can predict. Therefore, for a movie recommendation system, the NCF model with its low MSE and high R-squared would be considered the best-performing algorithm among the ones evaluated.

In order to enhance the performance of the Neural Collaborative Filtering (NCF) model for movie recommendations, several targeted strategies can be explored. First and foremost, feature augmentation offers the opportunity to incorporate additional user and movie features, such as demographic information or historical interactions, fostering a more nuanced understanding of user preferences. Moreover, the inclusion of contextual information, such as viewing context or time of day, can significantly improve the relevance of recommendations, contributing to a more personalized user experience. The exploration of hybrid models, combining collaborative filtering with content-based filtering, holds promise in capturing a broader range of user preferences and addressing limitations of individual recommendation approaches. Advanced embedding techniques, including matrix factorization or pre-trained embeddings, could be implemented to capture complex relationships in user-item interactions. Addressing temporal dynamics in user preferences, optimizing scalability, gathering real-time user feedback, and focusing on model explainability are additional avenues for refinement. Additionally, exploring cross-domain recommendations and implementing dynamic learning rates during training can further contribute to the adaptability and efficiency of the NCF model in providing accurate and personalized movie recommendations.

5.3 Case Study III (The TMDB Movie Dataset)

In the comparison of algorithms for the movie recommendation system, the hybird model emerged as the most promising, exhibiting the lowest Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) among the evaluated methods. The superior predictive accuracy of hybird model suggests its effectiveness in capturing user preferences and interactions. However, it's crucial to acknowledge that the choice of the optimal algorithm depends on various factors, including the dataset's characteristics and the specific nuances of user preferences. The decision-making process for selecting the most suitable algorithm should involve a deeper understanding of these factors.

In future work for the movie recommendation system using the hybrid model, which combines content-based and collaborative filtering techniques, should involve in-depth exploration of hyperparameters, including the weighting mechanism between components. Further refinement can be achieved through systematic parameter tuning and integration of advanced deep learning architectures within each component. Addressing scalability issues, exploring interpretability, and adapting the model to evolving user preferences are essential considerations. Additionally, conducting user studies for practical usability and feedback collection will contribute to the ongoing enhancement of hybrid recommendation systems.

The authors are grateful to the anonymous referee for a careful checking of the details and for helpful comments that improved this paper.

References

[1] S. Agrawal, "An Improved Approach for Movie Recommendation System," in *Proceedings of the International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*, 2017, pp. 1-5, doi: 10.1109/I-SMAC.2017.8058367.

[2] R. Subramaniam, R. Lee, and T. Matsuo, "Movie Master: Hybrid Movie Recommendation," in *2017 International Conference on Computational Science and Computational Intelligence*, IEEE, Mount Pleasant, MI, USA and Tokyo, Japan, p. 334, 2017.

[3] K.-Y. Jung, J.-H. Lee, and D.-H. Park, "A Hybrid Movie Recommender System Based on Neural Networks," *International Journal of Artificial Intelligence Tools*, vol. 16, no. 6, pp. 771-792, 2007.

[4] C. Christakou, S. Vrettos, and A. Stafylopatis, "A Hybrid Movie Recommender System Based on Neural Networks," *International Journal on Artificial Intelligence Tools*, vol. 16, no. 5, pp. 771-792, 2007, doi: 10.1142/S021821300700340X.

[5] P. Perny and J.-d. Zucker, "Preference-based Search and Machine Learning for Collaborative Filtering: the 'Film-Conseil' Movie Recommender System," *January*, 2001.

[6] M. Chenna Keshava, P. Narendra Reddy, S. Srinivasulu, and B. Dinesh Naik, "Machine Learning Model for Movie Recommendation System," *International Journal of Engineering Research & Technology (IJERT)*, vol. 9, no. 04, pp. 800-805, 2020.

[7] G. Miryala, R. Gomes, and K. R. Dayananda, "Comparative Analysis of Movie Recommendation System Using Collaborative Filtering in Spark Engine," *Journal of Global Research in Computer Science*, vol. 8, no. 10, pp. 1-14, 2017.

[8] H. Bhowmick, A. Chatterjee, and J. Sen, "Comprehensive Movie Recommendation System," *Journal of Advanced Research in Recommender Systems*, vol. 5, no. 2, pp. 112-127, 2019.

[9] D. K. Behera, S. Subhra, and P. K. Sethy, "Hybrid model for movie recommendation system using content K-nearest neighbors and restricted boltzmann machine," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 23, no. 1, pp. ab-cd, 2021, doi: 10.11591/ijeecs.v23.i1.ppab-cd.

[10] S. Roy, M. Sharma, and S. K. Singh, "Movie Recommendation System Using Semi-Supervised Learning," *Thakur College of Science and Commerce*, 2019.

[11] A. Roy and S. A. Ludwig, "Genre based hybrid filtering for movie recommendation engine," *Journal of Intelligent Information Systems*, vol. 56, pp. 485-507, 2021, Springer.

[12] L. Chen, J. Zhang, X. He, L. Nie, S. Liu, and T.-S. Chua, "A Comprehensive Survey on Cross-Domain Recommendation: Toward Context-Aware Recommendation," in *Proceedings of the 24th International Conference on World Wide Web*, 2017, pp. 5.

[13] S. Salloum and D. Rajamanthri, "Implementation and Evaluation of Movie Recommender Systems Using Collaborative Filtering," *Journal of Advances in Information Technology*, vol. 12, no. 3, pp. 192-194, 2021, doi: 10.12720/jait.12.3.192-194.

[14] M. C. Keshava, P. N. Reddy, S. Srinivasulu, and B. D. Naik, "Machine Learning Model for Movie Recommendation System," *International Journal of Engineering Research & Technology*, vol. 9, no. 4, pp. 804-805, 2020.

[15] F. Fouss and M. Saerens, "Evaluating performance of recommender systems: An experimental comparison," in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 2008.

[16] S. Agrawal and P. Jain, "An Improved Approach for Movie Recommendation System," in *International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*, IEEE, 2017, pp. 336-342.

[17] P. Kumar, S. G. Kibriya, Y. Ajay, and Ilampiray, "Movie Recommender System Using Machine Learning Algorithms," *Journal of Physics: Conference Series*, vol. 1916, pp. 012052, 2021, doi: 10.1088/1742-6596/1916/1/012052.

[18] A. Kaushik, S. Gupta, and M. Bhatia, "A Movie Recommendation System using Neural Networks," *International Journal of Advance Research, Ideas and Innovations in Technology*, vol. 4, no. 2, pp. 425-430, 2018.

[19] S. Sridevi and C. Murnal, "Implementation of Movie Recommendation System Using Machine Learning," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 6, no. 3, pp. 587-593, 2020, doi: 10.32628/CSEIT2063143.

[20] S. S. Choudhury, S. N. Mohanty, and A. K. Jagadev, "Multimodal trust based recommender system with machine learning approaches for movie recommendation," *International Journal of Information Technology*, vol. 13, no. 2, pp. 475-482, 2021, Springer.

[21] D. C. G. Putri, J.-S. Leu, and P. Seda, "Design of an Unsupervised Machine Learning-Based Movie Recommender System," *Symmetry*, vol. 12, p. 185, 2020.

[22] R. H. Singh, S. Maurya, T. Tripathi, T. Narula, and G. Srivastav, "Movie Recommendation System using Cosine Similarity and KNN," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 9, no. 5, 2020.

[23] N. Pavitha et al., "Movie recommendation and sentiment analysis using machine learning," *Global Transitions Proceedings*, vol. 3, pp. 279-284, 2022, Elsevier.

[24] K. Manoj, D. K. Yadav, A. Singh, and V. Kr, "A Movie Recommender System: MOVREC," *International Journal of Computer Applications*, vol. 124, pp. 7-11, 2015.

[25] J. Ananda Babu, D. R. Vinay, B. V. Kumaraswamy, and C. C. S. Basavaraddi, "Machine learning based recommendation system on movie reviews using KNN classifiers," *Journal of Physics: Conference Series*, vol. 1964, pp. 042081, 2021, doi: 10.1088/1742-6596/1964/4/042081.

[26] V. Subramaniyaswamy, R. Logesh, M. Chandrashekhar, A. Challa, and V. Vijayakumar, "A personalised movie recommendation system based on collaborative filtering," *Int. J. High Performance Computing and Networking*, vol. 10, nos. 1/2, pp. 54-63, 2017.

[27] M. F. Aljunid and D. H. Manjaiah, "An Efficient Deep Learning Approach for Collaborative Filtering Recommender System," *Procedia Computer Science*, vol. 171, pp. 829-836, 2020.

[28] GroupLens, "Movielens 20m dataset, Kaggle," 2018, https://www.kaggle.com/datasets/grouplens/movielens-20m-dataset/data, Accessed: 20 January 2024.

[29] R. Banik, "The Movies Dataset, Kaggle," 2017,https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset, Accessed: 20 January 2024.

[30] TMDb, "TMDB 5000 Movie Dataset," 2017, https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata, Accessed on 22 January 2024.

[31] W. Chen, F. Cai, H. Chen, and M. De Rijke, "Joint neural collaborative filtering for recommender systems," in *ACM Transactions on Information Systems (TOIS)*, vol. 37, no. 4, pp. 1–30, ACM New York, NY, USA, 2019.

[32] F. Anowar, S. Sadaoui, and B. Selim, "Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne)," *Computer Science Review*, vol. 40, pp. 100378, Elsevier, 2021.

[33] Rajeena PP, F., R. Orban, K. S. Vadivel, M. Subramanian, S. Muthusamy, D. S. A. Elminaam, A. Nabil, L. Abulaigh, M. Ahmadi, and M. A. S. Ali, "A novel method for the classification of butterfly species using pre-trained CNN models," *Electronics*, vol. 11, no. 13, pp. 2016, MDPI, 2022.

[34] D. S. A. AbdElminaam, N. ElMasry, Y. Talaat, M. Adel, A. Hisham, K. Atef, A. Mohamed, and M. Akram, "HR-chat bot: Designing and building effective interview chat-bots for fake CV detection," in *2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pp. 403–408, IEEE, 2021.

[35] M. A. S. Ali, R. Orban, R. R. Rajammal, S. Muthusamy, S. Subramani, K. Sekar, Rajeena PP, F., I. A. E. Gomaa, L. Abulaigh, and D. S. A. Elminaam, "A novel method for survival prediction of hepatocellular carcinoma using feature-selection techniques," *Applied Sciences*, vol. 12, no. 13, pp. 6427, MDPI, 2022.

[36] D. S. A. Abd Elminaam, A. Abdelaziz, G. Essam, and S. E. Mohamed, "AraFake: A deep learning approach for Arabic fake news detection," in *2023 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pp. 1–8, IEEE, 2023.

[37] D. S. A. AbdElminaam, N. Ahmed, M. Yasser, R. Ahmed, P. George, and M. Sahhar, "DeepCorrect: Building an Efficient Framework for Auto Correction for Subjective Questions Using GRU_LSTM Deep Learning," in *2022 2nd International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pp. 33–40, IEEE, 2022.

[38] D. S. A. AbdElminaam, M. M. Abouelwafa, A. E. Ibrahim, A. A. Saber, and L. Abualigah, "Forecasting Airlines Customer Demand: A Comparative Analysis of Time Series Models," in *2023 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pp. 1–7, IEEE, 2023.

[39] D. S. A. AbdElminaam, M. M. Madbouly, M. S. Farag, I. A. Gomaa, M. A. E. Zeid, and L. Abualigah, "ML_BrainDetection: An Intelligent Model for Brain Tumor Identification Using Machine Learning," in *2023 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pp. 1–7, IEEE, 2023.

[40] D. S. A. AbdElminaam, A. G. Fahmy, Y. M. Ali, O. A. D. El-Din, M. Heidar, et al., "DeepECG: Building an Efficient Framework for Automatic Arrhythmia classification model," in *2022 2nd International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pp. 203–209, IEEE, 2022.

[41] D. S. A. AbdElminaam, A. G. Fahmy, Y. M. Ali, O. A. D. El-Din, A. R. Aly, and M. Heidar, "ESEEG: An Efficient Epileptic Seizure Detection using EEG signals based on Machine Learning Algorithms," in *2022 2nd International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pp. 210–215, IEEE, 2022.

Deema AlSekait is a talented assistant professor with a strong information technology (IT) background. She received her PH.D. in information technology from Towson University, United States. Dr. AlSekait's extensive research portfolio covers a wide range of topics, from machine learning to health informatics to artificial intelligence and cloud computing. Additionally, Deema advocates for improving access to Science, Technology, Engineering, and Math careers. She is a shining example of an outstanding IT professional in an ever-changing field, as she perfectly combines her research expertise, pedagogical knowledge, and uncompromising commitment to social advancement. Deema's tireless efforts continue to inspire innovation, ensuring that the future of information technology remains bright and inclusive.

Ahmed Shdefat received his Ph.D. degree from the Software Engineering Department, School of Engineering, at Inje University, South Korea, in 2018. Previously, he served as a Research Assistant in the same department at Inje University. He is now an Assistant Professor in Computer Science at the College of Engineering and Technology, American University of the Middle East. His research interests encompass IoT infrastructure and architecture, ECG signal processing, bioinformatics, real-time processing, recognition of human activities, and e-health systems. He is also a reviewer for the IEEE Transactions on AgriFood Electronics and pervasive and mobile computing journals.

NOUR MOSTAFA , received a PhD degree from the Queen's University Belfast School of Electronics, Electrical Engineering and Computer Science, UK, in 2013. He previously worked as a software developer with Liberty Information Technology, USA/UK, and is currently an associate professor of computer science in the College of Engineering and Technology, American University of the Middle East. His current research interests include grid computing, large database management, artificial intelligence, machine learning, distributed computing, cloud, fog, and IoT computing. He has authored and co-authored many refereed journal articles, conference papers and book chapters. He is an active reviewer for many reputed international and IEEE journals and letters. Also, he has been selected as an International Steering Committee Member of many conferences, and he has joined the editorial board of international journals.

Alaa M. Hamdy Born on June 26, 2001, in Kuwait, Alaa Mohamed Mohamed Hamdy Ahmed is Computer Engineering , Ain Shams University, Faculty of Engineering , specializing in Data Science. Alaa envisions a future where she becomes a successful engineer, and her eagerness to learn and grow is reflected in the multiple internships she has undertaken for practical experience.These experiences, such as the Finyology Internship Program at NBE, Front End Development training with Sprints, DevOps training with WE, and a summer internship at AAIB, have significantly contributed to her professional development. Alaa's diverse experiences not only bolster her technical expertise but also serve as crucial stepping stones towards achieving her ambitious goals in the field of Computer Engineering. Looking ahead, Alaa aspires not only to be a successful engineer but also to make a positive impact and add value to the community through her skills and dedication. .

Diaa Salama is currently the Head of the Data Science Department at Misr International University, Egypt, on leave from Benha University since February 2020. He brings extensive experience in academia, having worked across several international and multicultural institutions. Dr. Salama has supervised numerous graduate projects, currently overseeing 13 master's and 3 Ph.D. students. His involvement in six research projects funded by King Faisal University, Saudi Arabia, in 2022, each supported by a grant of 40,000 Saudi Riyal, underscores his active research engagement. Dr. Salama has been recognized for his contributions to science and technology with several awards, including nominations for the Egyptian Encouragement Award for Advanced Science and Technology in 2021 and the Benha University Encouragement Award in 2019. He was also ranked among the top 2% of scientists worldwide by the Stanford University ranking in 2023. Additionally, he has consistently been ranked first at Benha University and highly among Egyptian and African universities by the AD Scientific Index from 2021 to 2024.His scholarly output includes over 150 research papers primarily in Q1 and Q2 journals, and he has presented at more than 30 international conferences. His publications have garnered over 2,670 citations. Dr. Salama is also an active member of several prestigious organizations, including the National Committee of Communications and Information Technology in Egypt and the International Federation for Information Processing.He holds a B.Sc. in Information Systems from Zagazig University (2004), and an M.Sc. (2009) and Ph.D. (2015) in Cloud Computing from Menoufia University. His research interests include optimization and artificial intelligence applications in the medical sector and renewable energy, focusing on feature selection, text mining, and object recognition.

Hana Fathi received her Ph.D. in computer science from the Faculty of Science, Menufia University, Egypt in 2022. She is an Assistant Professor at the Faculty of Information Technology, Applied Science private University. He has worked on several research topics. Hanaa has contributed more than 10+ technical papers in Feature Selection, classification, optimization, Machine learning, and web service in international journals. She attended the 2019 International Conference on Intelligent Systems and Advanced Computing Sciences (ISACS), in December 2019 in Taza, Morocco, and the fourth edition of the International Conference on Intelligent Systems and Computer Vision (ISCV 2020). She majors in machine learning, optimization, and medical application.