ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 14 (2024) No. 1, pp. 490-501
DOI:10.33168/JSMS.2024.0128

Optimizing a Personalized Movie Recommendation System with Support Vector Machine and Content-Based Filtering

Jonathan Leander, Arya Wicaksana*

Department of Informatics, Universitas Multimedia Nusantara, Tangerang 15810, Indonesia

arya.wicaksana@umn.ac.id

Abstract. Personalized movie recommendation systems have become increasingly popular in recent years. Support Vector Machine (SVM) and Content-Based Filtering (CBF) are popular techniques for building such systems. Cold start is a problem in any recommendation system requiring a specific method to address. This paper discusses the use of SVM and CBF in building a personalized movie recommendation system. The cold start problem is addressed by jump-starting the recommendation system. The recommendation system and web services are developed as a web-based application as proof of concept and feasibility. The application interface for recommendation is in Top-N List, and the technology used to provide recommendations is Attribute-Based Recommendation with Organic Navigation. Movie datasets are obtained from the Movie Database (TMDB). The hyperparameter tuning process yields the best accuracy of 88.5% with RBF kernel function (C=1 and gamma=10) on 250 movie datasets. Further analysis shows that movie trailers and posters are two attributes that affect user preferences on liking/disliking a movie.

Keywords: CBF, movie recommendation system, personalization, SVM

1. Introduction

Personalized movie recommendation systems are important to provide movie suggestions based on an individual's viewing history and preferences, making the experience more enjoyable for the user. Movie recommendation systems are designed to help users discover new movies they may be interested in watching based on their past preferences (Chen et al., 2021). There are several applications of movie recommendation systems, including personalized movie recommendations, i.e., Netflix. Netflix is one of the most popular streaming services globally. As of 2023, it has over 220 million subscribers and the highest number of subscribers in 78 countries, making it the largest streaming platform in the world (Fernandes, 2023). The popularity of Netflix can be attributed to its extensive library of original content, convenient user interface, and affordable pricing (Hansen, 2017).

Netflix uses a hybrid recommendation system combining collaborative and content-based filtering to recommend movies and TV shows to its users. In collaborative filtering, Netflix uses the data from millions of its subscribers to make recommendations based on the viewing patterns of similar users. In content-based filtering, the system analyzes the attributes of the movies, such as genre, cast, director, and more, to make recommendations based on the similarity between the movies a user has liked in the past and the content of new movies. In addition, Netflix also uses other algorithms and techniques, such as matrix factorization, deep learning, and reinforcement learning to enhance its recommendation system and provide a more personalized experience for its users (Jenkin, 2021).

Support Vector Machine (SVM) and Content-Based Filtering are popular techniques for personalized movie recommendation systems (Pavitha et al., 2022). Support Vector Machine (SVM) is a machine learning algorithms for binary classification problems (Gholami & Fakhari, 2017). The motivation for using SVMs in personalized movie recommendation systems is to predict whether users will like a particular movie based on their past preferences. In this context, the SVM algorithm trains a model on the historical data of a user's preferences. Each movie is represented as a set of features, such as the genre, cast, director, and year of release. The SVM can handle large datasets and high-dimensional feature spaces (Cervantes et al., 2008; Suykens, 2009). This is important in movie recommendation systems, as there are typically many features associated with each movie. The SVM algorithm can effectively handle these high-dimensional feature spaces and provide accurate predictions.

Content-Based Filtering (CBF) recommends items that have similar characteristics as items used by the user in the past (Tewari, 2020). CBF focuses on the content of the movies, such as their genre, cast, director, and other attributes. The motivation for using Content-Based Filtering in personalized movie recommendation systems is to provide recommendations based on the content of the movies. In this method, the recommendations are generated based on the similarity between the content of the movies a user has liked and the content of new movies. The algorithm calculates the similarity between the movies by comparing their content-based attributes, such as genre, cast, director, and other features. The movies with the highest similarity scores are then recommended to the user.

This study contributes to using SVM and CBF for personalized movie recommendation systems with real-time adaptability to new movies and addressing the cold-start problem. The design and implementation of the personalized movie recommendation system involve hyperparameter tuning the SVM and jump-starting the system for first-time users. Testing and evaluation involve evolving user preferences and large and diverse movie catalogs, a combination of multiple factors, and the cold start problem. The Movie Database (TMDB) is used as the data set for training and testing purposes of the SVM. Accuracy and F1-Score of the system are observed from the experiments.

The rest of this paper is organized as follows. Section 2 describes the preliminaries related to the study. Section 3 describes the research methods. Section 4 explains the results and discussion. Finally, Section 5 concludes this paper with some suggestions for future work.

2. Preliminaries

2.1. Related Work

(Pavitha et al., 2022) performed sentiment analysis on movie reviews using machine learning to enhance the user experience on movie recommendation systems. Using SVM and Naïve Bayes has proven to improve the system's accuracy with SVM outweighs Naïve Bayes for the sentiment analysis process. Another work in (Salmani & Kulkarni, 2021) proposed a hybrid movie recommendation system using machine learning, which yields the best performance in terms of RMSE compared to CBF, collaborative-based filtering, single value decomposition (SVD), and SVD++. The implementation of SVM for facial skin type identification achieved an F1-score of 0.85 (Utami et al., 2020). In (Kristiyanti & Sri Hardani, 2023), the SVM method accuracy for sentiment analysis overcame Naïve Bayes and LSTM with 84.89% compared to 84.65% and 82.97%, respectively.

(Sharma & Dutta, 2020) and (Jayalakshmi et al., 2022) surveyed movie recommender systems covering concepts, methods, challenges, and future directions. Among the five identified problems of movie recommender systems, the cold-start problem and accuracy are addressed in this study. (Vilakone et al., 2018) proposed an improved k-clique method that delivers higher accuracy compared to k nearest neighbor, maximal clique method, and k-clique method with collaborative filtering. The best method in terms of mean absolute percentage error was found with k = 11 and rated at least 200 movies. The k-clique method involves finding dense subgraphs of a given size (k) in a graph which can become computationally expensive and challenging to scale for large datasets compared to SVM. SVM can also effectively work with sparse data by using appropriate kernel functions and regularization techniques, including encouraging diversity in recommendations through custom loss functions or ensemble methods.

2.2. Personalized Movie Recommendation System

A personalized movie recommendation system is a software tool designed to provide users with personalized suggestions for movies or TV shows based on their preferences and viewing history (Airen & Agrawal, 2023; Behera & Nain, 2023; Zhang & Zhang, 2022). The system utilizes machine learning algorithms to analyze user data, including ratings, watch history, and genre preferences, to generate a tailored list of recommendations. By delivering relevant and appealing recommendations to each user, these systems enhance the viewing experience and help users discover new content they might enjoy. There are several open issues in personalized movie recommendation systems, as follows.

- Cold-start problem: Recommendation systems often struggle to provide accurate recommendations for new users with limited data or who have not explicitly stated their preferences (Lika et al., 2014).
- Scalability: Handling many users and items can be challenging for recommendation systems, leading to slow performance and decreased accuracy (Lee & Lee, 2019).
- Diversity and novelty: Recommendation systems tend to recommend popular items and can struggle to introduce new or less well-known items to users (Berbague et al., 2021).
- Data quality and privacy: The quality and accuracy of the data used to train recommendation systems are crucial, and user data privacy must also be protected (Yan et al., 2022).
- Explainability and trust: Users may not trust recommendations that they do not understand, and there is a need for recommendation systems to be more transparent and explainable (Wei et al., 2023).
- Context-awareness: Recommendation systems often lack context-awareness, such as the time of day, the user's location, or their mood, which can impact their preferences (Stitini et al., 2022).

Determining the best performance model for a personalized movie recommendation system is complex. It depends on several factors, such as the data available, the target user group, and the specific use case. Several models have been shown to achieve good performance in practice:

- Collaborative Filtering (CF): CF is a classic technique based on finding similar users or items and using that information to make recommendations. It is simple to implement and has performed well in many cases (Behera & Nain, 2023).
- Matrix Factorization (MF): MF is a type of CF that factorizes the user-item matrix into lower-dimensional latent representations. It can handle large and sparse data sets and is often used for personalized recommendations (Wang et al., 2023).
- Deep Learning: Neural networks and deep learning techniques, such as autoencoders and attention-based models, have performed well for recommendation tasks. These models can handle large amounts of data and capture complex non-linear relationships between users and items (Li et al., 2023).
- Hybrid Models: Combining multiple models can often result in better performance than using a single model. For example, incorporating the sentiment of product reviews for a better recommendation process (Elahi et al., 2023).

2.3. Support Vector Machine

The machine learning algorithm Support Vector Machine (SVM) is useful for both classification and regression analysis. Finding the ideal decision boundary to divide various data classes is the goal of SVM. SVMs look for the hyperplane in classification issues that optimizes the margin between the various data classes. The margin is the separation between the nearest data points from each class and the hyperplane. The hyperplane that maximizes the margin performs the classification rule generalization to unseen data the best. The hyperplane is therefore the most resistant to noise and outliers (Awad & Khanna, 2015). SVMs look for the hyperplane that minimizes the prediction error and best matches the data in regression tasks (Collobert & Bengio, 2001). By utilizing the kernel method, which moves the input data into a higher-dimensional space where a linear decision boundary may be located, SVMs can also be employed for non-linear classification and regression issues (Fávero et al., 2023).

A linear decision boundary can be located in a higher-dimensional space where the input data is mapped by the kernel function (Awad & Khanna, 2015). Without explicitly changing the input data points, the kernel function computes the dot product between the data points in the higher-dimensional space. As a result, the original input space's non-linear decision boundary can be discovered via SVM. The three varieties of kernel functions used in this research are (Awad & Khanna, 2015; scikit-learn developers, 2023a):

1. Polynomial Kernel: $k(x_a, x_b) = (x_1^T x_2 + c)^d$
2. Gaussian (Radial Basis Function) Kernel: $k(x_a, x_b) = \sigma^2 \exp\left(-\frac{||x_a - x_b||^2}{2l^2}\right)$
3. Sigmoid Kernel: $k(x, y) = \tanh\left(\gamma . x^T y + r\right)$

The SVM algorithm's performance can be considerably impacted by the kernel function's parameters and choice. The characteristics of the problem and the input data should be taken into consideration while selecting the kernel function and its parameters. The following kernel parameters were tuned in this study (scikit-learn developers, 2023b):

1. C: The regularization parameter that regulates the compromise between increasing the margin and reducing the classification error.
2. Gamma: The variable that affects how the decision boundary in higher-dimensional space is shaped. A smoother choice border is produced by a smaller gamma value, whereas a more significant gamma value produces a more complex decision boundary.
3. Degree: The degree of the polynomial function used in the polynomial kernel.
4. Coefficient: The coefficient of the polynomial function used in the polynomial kernel.

2.4. Content-Based Filtering

Users are given recommendations for things through content-based filtering systems based on their prior interactions with comparable items. In content-based filtering, the system analyzes the attributes or characteristics of items a user has liked or interacted with and recommends other items with similar attributes (Google, 2022). In a movie recommendation system, when a user likes action movies, the system would recommend other action movies based on similar attributes, such as high-intensity action sequences. Content-based filtering in this study comprised of the following steps:

1. Feature Extraction: The algorithm pulls out pertinent information about the items from the data, such as their genre, actor, director, year of release, etc.
2. Item Profile Creation: The system builds an item profile for each item based on its characteristics or qualities. The item's features are described in the profile, which can also be used to contrast it with other products.
3. User Profile Creation: Based on interactions with items, the system builds a user profile for each user. The user's preferences are described in the profile, which can also be used to suggest products that are similar to those the user has already engaged with.
4. Similarity Calculation: Using the traits or qualities of the user and object profiles, the system determines how similar the two profiles are. As a result, the system can determine which goods are closest to the user's tastes.
5. Recommendation Generation: The algorithm generates a list of recommendations that are most similar to the user's preferences based on the similarity calculation.

3. Methods

The main features of the personalized recommendation system developed in this study are (i) a Graphical user interface with the top-n list and item-to-item correlation recommendation display; (ii) system access through organic navigation; (iii) real-time movie dataset fetched by the system from The Movie Database (TMDB) API; (iv) local database for the knowledge subsystem required in recommendation process; (v) building up the knowledge set (jump-start) to avoid the cold-start problem; and (vi) recommendation engine fine-tuning on a routine basis. The web service flowchart of the personalized movie recommendation system is displayed in Figure 1.

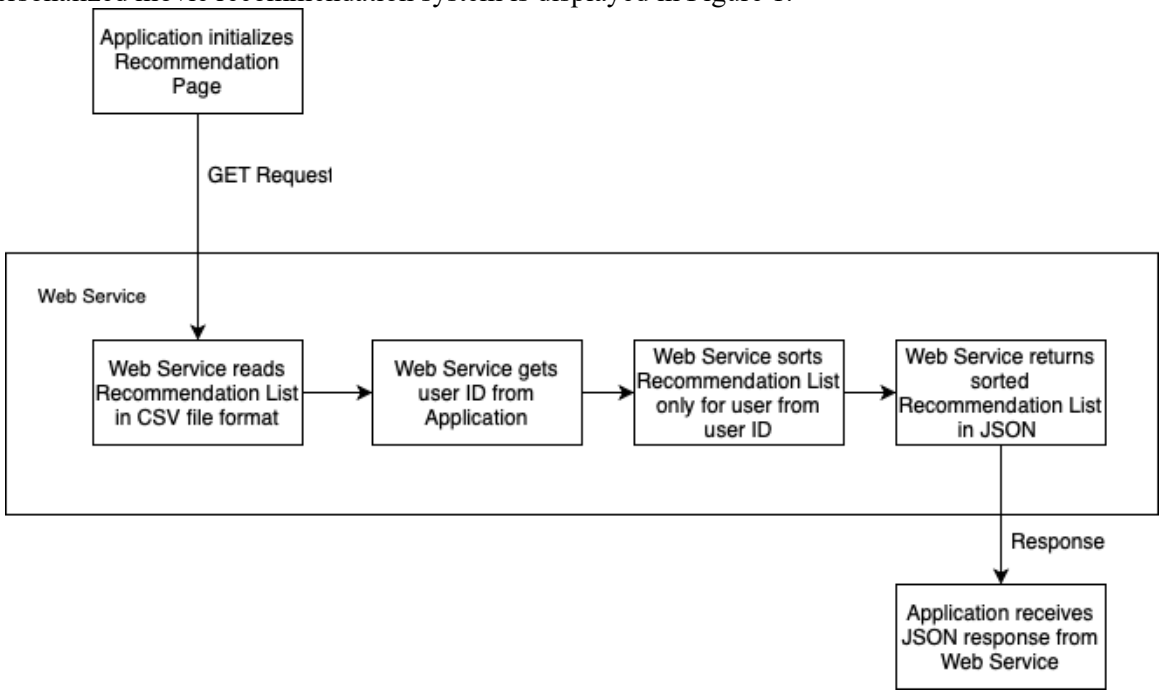

Fig. 1: Application web service flowchart.

The recommendation process starts with user inputs of personally liked movies to jump-start the system. This design aims to solve the cold start problem by encouraging the user to set up the knowledge base of the recommendation system. The implementation of this step involves up to 250 personally liked movies to test and evaluate the performance of the recommendation part. The process continues to start with the web service flowchart shown in Figure 1, where the application requests for recommendation list using the HTTP GET method. The recommendation page implements the SVM and CBF for providing the recommended movies list. Implementing the recommendation feature as web services allows interoperability of this recommendation system with other front-end applications. The implementation of the machine learning part (SVM) uses Anaconda Navigator 2.2.0 with Conda version 22.9.0, Jupyter Notebook v6.4.8, Google Colab Pro version 2022/10/21, Python 3.9.12, and Flash Python Framework v1.1.2. The rest of the recommendation system is built with Microsoft Visual Studio Code 2019 1.73.1, XAMPP Control Panel v3.3.0, phpMyAdmin v.5.2.0, and Apache v2.4.53. Figure 2 shows the complete recommendation system workflow.

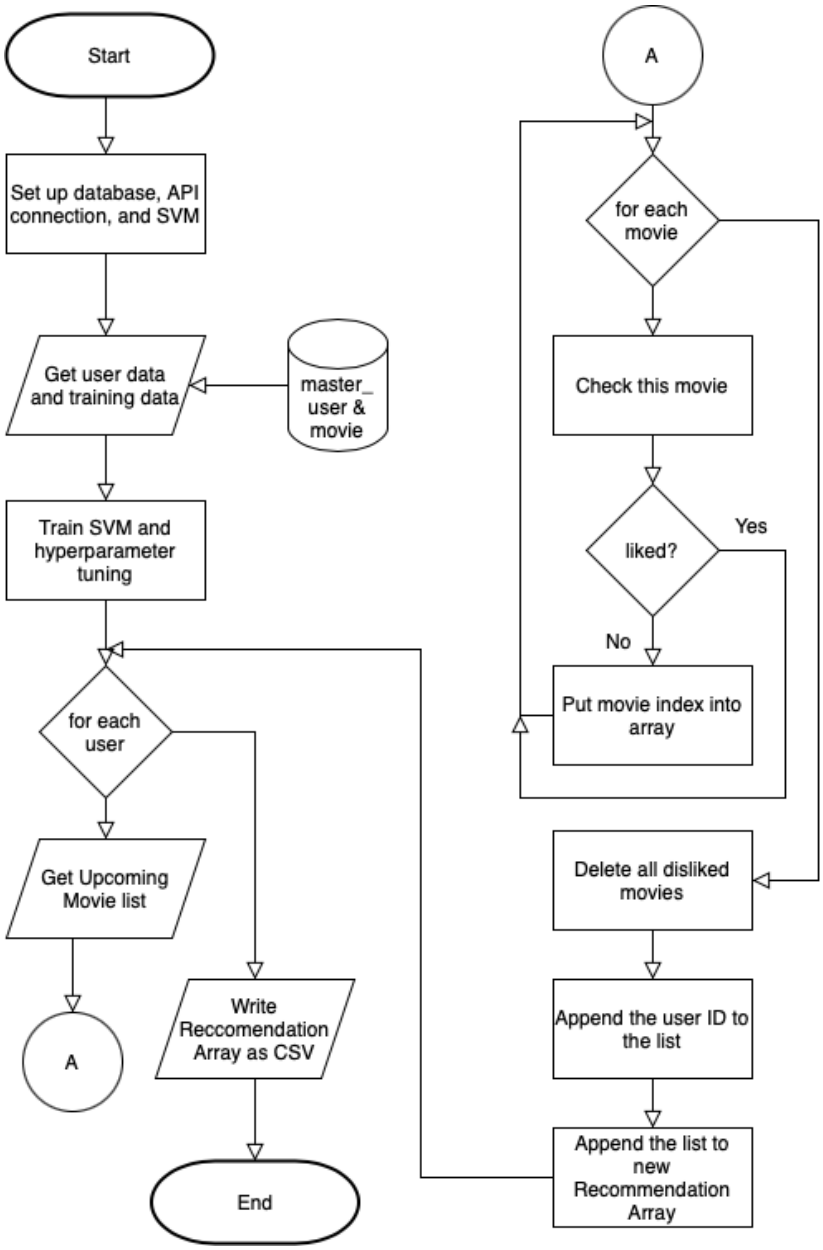

Fig. 2: Recommendation system flowchart with SVM and CBF.

The recommendation system comprises a database, web services, and a recommendation module

(SVM and CBF). The jump-start procedure explained earlier in this section provides the system with information regarding user preferences for the movies. Based on this information, the system retrieves movies from the movie database and applies CBF on the data to filter out unwanted movies. It then performs hyperparameter tuning of the SVM based on the provided personally liked movies. Hyper Parameter Tuning is done by using GridSearchCV to find the best combination of parameters and using a classification report to evaluate the accuracy value and F1-Score, which is done repeatedly by differentiating the amount of training data used: 10, 20, 50, 100, 200, 250 with 80% split for training and 20% split for testing. Hyperparameter tuning uses GridSearchCV with the Kernel Function parameter Polynomial, RBF, and Sigmoid with the range of C and gamma values used are 10^{-3} to 10^{3}.

As shown in Figure 2, the system periodically updates its movie database by fetching real-time movie data from the TMDB. It then fine-tunes the SVM to increase further the recommendation system's accuracy, precision, and recall. The test and evaluation of the recommendation system started with three scenarios. In each scenario, 250 movies are reviewed by different users to produce the knowledge base. After the recommendation system produces a list of film recommendations for each user, each list will be evaluated using accuracy metrics and an F1-Score. This list of upcoming movies for testing and evaluation shown in Table 1 is retrieved from TheMovieDatabase API on December 12th, 2022.

Table 1: The upcoming movies list.

Movie Title	Release Date
Avatar: The Way of the Water	14 Dec 2022
M3GAN	6 Jan 2023
Babylon	23 Dec 2022
A Man Called Otto	25 Dec 2022
Operation Fortune: Ruse de Guerre	5 Jan 2023
The Pale Blue Eye	23 Dec 2022
I Wanna Dance with Somebody	23 Dec 2022
Project X-Traction	12 Dec 2022
Last Weekend	12 Dec 2022

4. Results and Discussion

The hardware specifications used for the development including testing and evaluation the system are Intel® CoreTM i3-7100 Processor (3M Cache) @3.90 GHz with 8GB of RAM and NVIDIA GeForce GTX 1050 TI. The software used are Windows 10 64-bit Operating System, Anaconda Naviagtor 2.2.0 with conda version 22.9.0, Jupyter Notebook v6.4.8, Google Chrome version 107.0.5304.121, Microsoft Visual Studio Code 2019 1.73.1, XAMPP Control Panel v3.3.0, phpMyAdmin v5.2.0, Google Colab Pro version 2022/10/21, Apache v2.4.53, Python 3.9.12, and Flask Python Framework v1.1.2. The implementation of the recommendation system interface describes the workings of the recommendation system that has been built using the Ionic Framework.

The implementation of the graphical user interface of the personalized movie recommendation system is displayed in Figures 3-4. The user interface design adopts organic navigation suitable for handheld mobile devices. The organic navigation allows ease of access to the main features of the system: popular movies, search movies, recommendation page (For Me), and upcoming movies, including the detailed movie page. The Popular Movies page obtained a list of popular movies from the TMDB. The recommendation system updates the inference model every time user likes a movie, and this change is updated immediately and can be seen on the recommendation page (For Me). The Upcoming Movies page retrieved a list of upcoming movies from the TMDB. The user could browse through upcoming movies and click to see more details on the movie.

The implementation of the recommendation subsystem begins by performing Hyper Parameter

Tuning on SVM and then implementing SVM on the recommendation system. The results of the hyperparameter tuning are summarized in Table 2. The best accuracy obtained for the inference model is 0.885 on 250 movies with kernel function = RBF, C = 1, and gamma = 10.

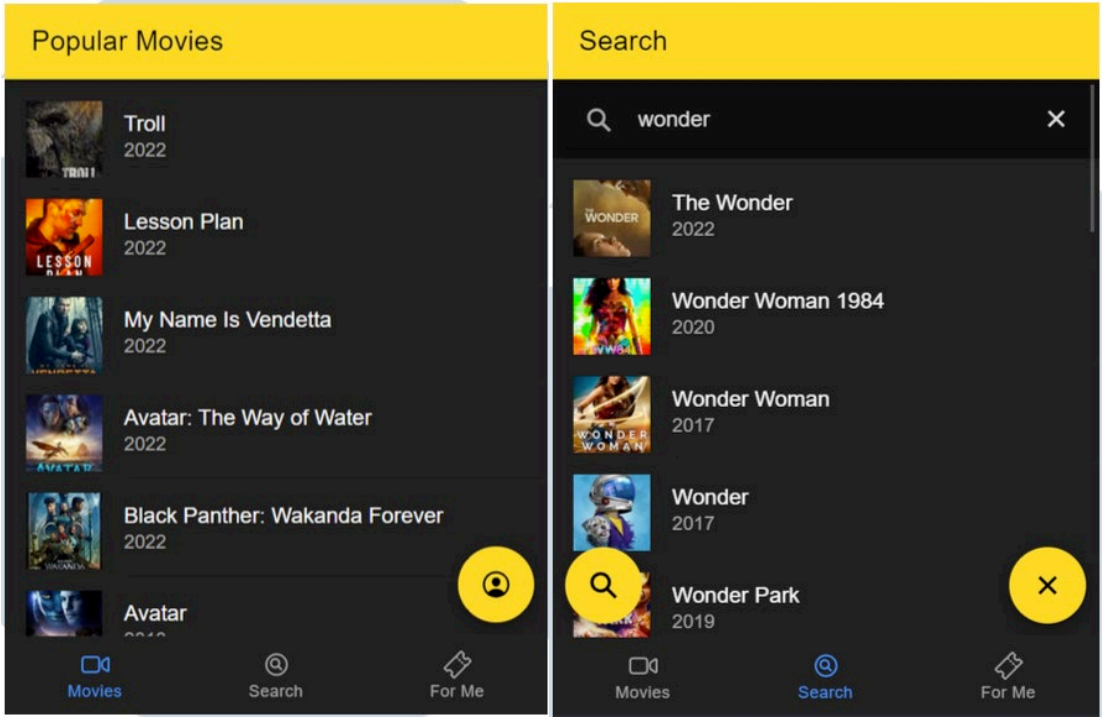

Fig. 3: Popular movies page (left). Search movie page (right).

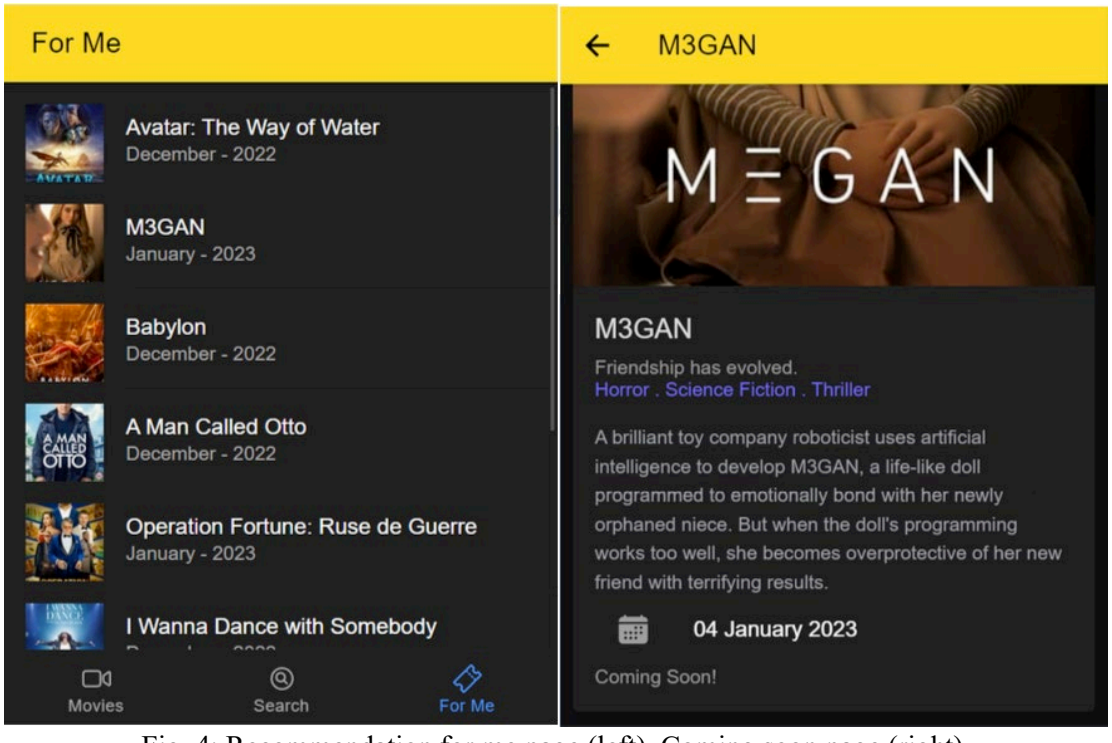

Fig. 4: Recommendation for me page (left). Coming soon page (right).

Table 2: Hyperparameter results.

No.	Dataset Size	Kernel Function	C	Gamma	Accuracy
1	14 Dec 2022	Polynomial	0.001	10	0.86
2	6 Jan 2023	RBF	100	0.1	0.86
3	23 Dec 2022	Sigmoid	1	0.1	0.825
4	25 Dec 2022	RBF	1	1	0.775
5	5 Jan 2023	RBF	1	1	0.868
6	23 Dec 2022	RBF	1	10	0.885

Based on the results of the Hyper Parameter tuning that has been done, SVM implementation in the recommendation system begins by declaring SVM, Kernel Function, and other Hyper Parameters. The movie dataset is separated into variable x as the Input Set, which contains the genres column, and variable y as the Output Set, which contains the liked column for training the SVM. User data registered through the application is required to recommend movies to each user. API Request from TheMovieDatabase API retrieves a list of upcoming movies. Predictions are made on each user's relationship with each movie within the list of upcoming movies, and movies predicted to be disliked by the user are removed from the list.

Further tests are conducted on the inference model to measure the accuracy and F1-Score of the personalized movie recommendation system. Three case studies are developed involving three different users with the list of upcoming movies given in Table 1. Each user is asked to either like or dislike 250 movies which is part of the jump-starting process to address the cold-start problem in this study, and the upcoming movies list in Table 1 is used as the prediction target against each user. The accuracy and F1-Score of the three case studies are displayed in Figure 5.

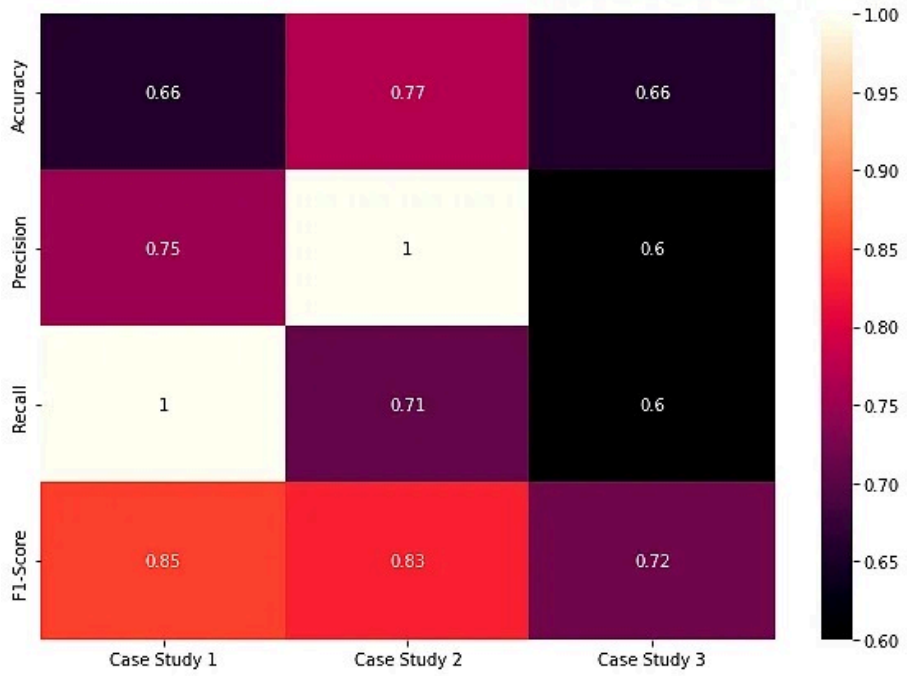

Fig. 5: System performance: accuracy and F1-score.

The case studies show that the actual system performance in accuracy is lower than the inference model (hyperparameter-tuned model). Further evaluation is carried on through interviews with the three users. It is found that the movie trailer and poster play a significant role in affecting user decisions towards liking or disliking the movies. The two attributes are not considered in the initial design of the personalized recommendation system and require further study to enhance the system. The limited

number of users for training and testing the model can affect the overall system performance regarding accuracy and F1-score. In addition to the jump-starting procedure, users must correctly choose the movie they like. The results obtained in this study are arguably better than similar studies on SVM applications mentioned in the related work subsection.

5. Conclusions

This study contributes to using Support Vector Machine (SVM) and Content-Based Filtering (CBF) for personal movie recommendation systems. It also addressed the cold-start problem by jump-starting the recommender engine at the beginning. Users are asked to choose the preferred movies from several movies, with every genre presented equally at the beginning of using the system, as part of jump-starting the system, which shows significant improvement for better accuracy and F1-score. Real-time movie datasets are obtained from the Movie Database (TMDB) for training and testing purposes of the recommendation system. Through testing and evaluation, it is found that the best parameters for the SVM are the RBF kernel function with a C value of 1 and a gamma value of 10, giving the best accuracy of 88.5%.

Further tests and analysis of the system's performance show that movie trailers and posters affect user preferences for liking or disliking a movie. This information is concluded from interviews with each user in the case studies. In addition to the jump-starting process, that takes time and causes inconvenience to some users. Suggestions for the development of this study are as follows.

1. Application of multi-threading and incremental learning in the recommendation system so that the recommendation system can continue to develop efficiently as the movie data increases.
2. Adding a movie trailer and poster in the training process for the SVM and CBF to boost system performance.

Acknowledgements

Universitas Multimedia Nusantara fully supports this study. The authors thank the reviewers for the feedback to enhance the quality of this paper.

References

Airen, S., & Agrawal, J. (2023). Movie Recommender System Using Parameter Tuning of User and Movie Neighbourhood via Co-Clustering. *Procedia Computer Science*, *218*, 1176–1183. https://doi.org/https://doi.org/10.1016/j.procs.2023.01.096

Awad, M., & Khanna, R. (2015). Support Vector Machines for Classification. In *Efficient Learning Machines* (pp. 39–66). https://doi.org/https://doi.org/10.1007/978-1-4302-5990-9_3

Behera, G., & Nain, N. (2023). Collaborative Filtering with Temporal Features for Movie Recommendation System. *Procedia Computer Science*, *218*(1366–1373). https://doi.org/https://doi.org/10.1016/j.procs.2023.01.115

Berbague, C. E., Karabadji, N. E., Seridi, H., Symeonidis, P., Manolopoulos, Y., & Dhifli, W. (2021). An overlapping clustering approach for precision, diversity and novelty-aware recommendations. *Expert Systems with Applications*, *177*. https://doi.org/https://doi.org/10.1016/j.eswa.2021.114917

Cervantes, J., Li, X., Yu, W., & Li, K. (2008). Support vector machine classification for large data sets via minimum enclosing ball clustering. *Neurocomputing*, *71*(4), 611–619. https://doi.org/DOI: 10.1016/j.neucom.2007.07.028

Chen, Y.-L., Yeh, Y.-H., & Ma, M.-R. (2021). A movie recommendation method based on users' positive and negative profiles. *Information Processing & Management*, *58*(3). https://doi.org/https://doi.org/10.1016/j.ipm.2021.102531

Collobert, R., & Bengio, S. (2001). SVMTorch: Support Vector Machines for Large-Scale Regression Problems. *Journal of Machine Learning Research*, *1*, 143–160.

Elahi, M., Kholgh, D. K., Kiarostami, M. S., Oussalah, M., & Saghari, S. (2023). Hybrid recommendation by incorporating the sentiment of product reviews. *Information Sciences*, *625*, 738–756. https://doi.org/https://doi.org/10.1016/j.ins.2023.01.051

Fávero, L. P., Belfiore, P., & Souza, R. de F. (2023). Support vector machines. In *Data Science, Analytics and Machine Learning with R* (pp. 323–370). Academic Press. https://doi.org/https://doi.org/10.1016/B978-0-12-824271-1.00032-9

Fernandes, F. (2023). *Mapped: The Most Popular Video Streaming Service by Country*. Visual Capitalist. https://www.visualcapitalist.com/cp/mapped-the-most-popular-video-streaming-service-by-country/

Gholami, R., & Fakhari, N. (2017). Support Vector Machine: Principles, Parameters, and Applications. In *Handbook of Neural Computation* (pp. 515–535). Academic Press. https://doi.org/https://doi.org/10.1016/B978-0-12-811318-9.00027-2

Google. (2022). *Content-based Filtering*. Recommendation Systems. https://developers.google.com/machine-learning/recommendation/content-based/basics

Hansen, B. (2017). DISRUPTIVE INNOVATION: A CASE STUDY ON HOW NETFLIX IS TRANSFORMING THE LIVING ROOM [Copenhagen Business School]. In *Master's Thesis in Cand.Soc. Management of Creative Business Processes*. https://research-api.cbs.dk/ws/portalfiles/portal/60751689/248533_Bianca_Hansen_Masters_Thesis_CBP_2017_Case_Study_on_Netflix.pdf

Jayalakshmi, S., Ganesh, N., Čep, R., & Senthil Murugan, J. (2022). Movie Recommender Systems: Concepts, Methods, Challenges, and Future Directions. *Sensors*, *22*(13), 4904. https://doi.org/10.3390/s22134904

Jenkin, C. (2021). *How Netflix uses recommender systems*. Machine Learning Group. https://reflect.ucl.ac.uk/nsci0010-2021-class-blog/2021/02/24/how-netflix-uses-recommender-systems/

Kristiyanti, D. A., & Sri Hardani. (2023). Sentiment Analysis of Public Acceptance of Covid-19 Vaccines Types in Indonesia using Naïve Bayes, Support Vector Machine, and Long Short-Term Memory (LSTM). *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *7*(3), 722–732. https://doi.org/10.29207/resti.v7i3.4737

Lee, H., & Lee, J. (2019). Scalable deep learning-based recommendation systems. *ICT Express*, *5*(2), 84–88. https://doi.org/https://doi.org/10.1016/j.icte.2018.05.003

Li, B., Li, G., Xu, J., Li, X., Liu, X., Wang, M., & Lv, J. (2023). A personalized recommendation framework based on MOOC system integrating deep learning and big data. *Computers and Electrical Engineering*, *106*. https://doi.org/https://doi.org/10.1016/j.compeleceng.2022.108571

Lika, B., Kolomvatsos, K., & Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*, *41*(4), 2065–2073. https://doi.org/https://doi.org/10.1016/j.eswa.2013.09.005

Pavitha, N., Pungliya, V., Raut, A., Bhonsle, R., Purohit, A., Patel, A., & Shashidhar, R. (2022). Movie recommendation and sentiment analysis using machine learning. *Global Transitions Proceedings*, *3*(1), 279–284. https://doi.org/https://doi.org/10.1016/j.gltp.2022.03.012

Salmani, S., & Kulkarni, S. (2021). Hybrid Movie Recommendation System Using Machine Learning. *2021 International Conference on Communication Information and Computing Technology (ICCICT)*.

https://doi.org/10.1109/ICCICT50803.2021.9510058

scikit-learn developers. (2023a). *1.4. Support Vector Machines*. Scikit Learn. https://scikit-learn.org/stable/modules/svm.html#kernel-functions

scikit-learn developers. (2023b). *sklearn.svm.SVC*. Scikit Learn. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Sharma, N., & Dutta, M. (2020). Movie Recommendation Systems. *Proceedings of the 8th International Conference on Computer and Communications Management*, 59–62. https://doi.org/10.1145/3411174.3411194

Stitini, O., Kaloun, S., & Bencharef, O. (2022). Integrating contextual information into multi-class classification to improve the context-aware recommendation. *Procedia Computer Science*, *198*(311–316). https://doi.org/https://doi.org/10.1016/j.procs.2021.12.246

Suykens, J. A. K. (2009). Kernel Methods. In S. D. Brown, R. Tauler, & B. Walczak (Eds.), *Comprehensive Chemometrics* (pp. 437–451). Elsevier. https://doi.org/https://doi.org/10.1016/B978-044452701-1.00059-4

Tewari, A. S. (2020). Generating Items Recommendations by Fusing Content and User-Item based Collaborative Filtering. *Procedia Computer Science*, *167*, 1934–1940. https://doi.org/https://doi.org/10.1016/j.procs.2020.03.215

Utami, M. T., Young, J. C., & Wicaksana, A. (2020). Implementation of Support Vector Machine Algorithm for Identifying Facial Skin Types. *TEST Engineering & Management*, *83*.

Vilakone, P., Park, D.-S., Xinchang, K., & Hao, F. (2018). An Efficient movie recommendation algorithm based on improved k-clique. *Human-Centric Computing and Information Sciences*, *8*(1), 38. https://doi.org/10.1186/s13673-018-0161-6

Wang, Y., Gao, M., Ran, X., Ma, J., & Zhang, L. Y. (2023). An improved matrix factorization with local differential privacy based on piecewise mechanism for recommendation systems. *Expert Systems with Applications*, *216*. https://doi.org/https://doi.org/10.1016/j.eswa.2022.119457

Wei, T., Chow, T. W. S., Ma, J., & Zhao, M. (2023). ExpGCN: Review-aware Graph Convolution Network for explainable recommendation. *Neural Networks*, *157*, 202–215. https://doi.org/https://doi.org/10.1016/j.neunet.2022.10.014

Yan, D., Zhao, Y., Yang, Z., Jin, Y., & Zhang, Y. (2022). FedCDR: Privacy-preserving federated cross-domain recommendation. *Digital Communications and Networks*, *8*(4), 552–560. https://doi.org/https://doi.org/10.1016/j.dcan.2022.04.034

Zhang, Y., & Zhang, L. (2022). Movie Recommendation Algorithm Based on Sentiment Analysis and LDA. *Procedia Computer Science*, *199*, 871–878. https://doi.org/https://doi.org/10.1016/j.procs.2022.01.109