A movie recommendation system based on machine learning techniques

Anand Prakash Srivastava[1], Dr. Sakuldeep Singh[2]
[1,2] *Computer Science & Engineering, Sanskriti University, Mathura*

***Abstract*—This study presents a movie recommendation system tailored to the cosine similarity for recommendations. From the tmdb dataset, a random movie is chosen, and ten similar movies are recommended using cosine similarity. The system's effectiveness was evaluated using two machine learning algorithms: Naive Bayes (Gaussian, Multinomial, Bernoulli) and Support Vector Machine (SVM) with linear and radial basis function (rbf) kernels. Models were trained on datasets comprising 75%, and 80% of the available data, and their performance was assessed. Results indicated that the SVM method, particularly with the linear kernel, achieved the highest accuracy, while the Naive Bayes showed the lowest accuracy. The SVM algorithm's consistent and superior performance highlights its suitability for this recommendation system, whereas Naive Bayes was less effective for this application.**

***Index Terms*— Cosine Similarity, Movie recommendation system, Naive Bayes, Support Vector Machine**

I. INTRODUCTION

Recommendation systems [1] are adopted in various sectors like e-commerce, retail banking and entertainment. The main goal of these systems is to provide recommendations to users based on their data, which are collected constantly and analyzed. The most popular methods for recommendation system [2] are Content-based Filtering (CBF), Collaborative Filtering (CF) and Hybrid Filtering. With the use of CBF (Content based filtering) technique [3] which checks about the features of each item and suggests other items that have similar attributes. By exploring the correspondence between users and items, CF [4, 5] improves some drawbacks of CBF and provides suggestions. It uses the data of user's past selection and other likeminded users choice to offer personalized recommendations. Many existing recommendation systems (RS) use a hybrid-filtering (HF) technique [6] that mixes the strengths of Content-Based filtering (CBF) and Collaborative Filtering (CF). A film rides on audiences feedback. Other users rely so much on these reviews while making their own choices. People tend to be more likely to choose a movie that has been well-received by the majority, rather than one that has been mostly disliked. Considering these reviews, excluding those that contain misleading information, also adds to the intricacy of decision-making. There is a potential solution to this problem through sentiment analysis. Utilizing Natural Language Processing (NLP), Sentiment Analysis [7] allows for the extraction of information from textual sources and the classification of statements, words, or documents as positive or negative. Understanding the author's perspective and sharing one's own experiences can be highly valuable. Opinion mining utilizes the principles of data mining to extract and categorize the viewpoints expressed in diverse online forums or venues. This facilitates a more comprehensive comprehension of the user's sentiments or emotions pertaining to a specific topic. [8].

This paper proposed movie recommendation system based on cosine similarity. The recommendation system is then evaluated using two different machine learning algorithms namely Naive Bayes (NB) using gaussian nb, multinomial nb and bernoulli nb and Support Vector Machine (SVM) using linear and rbf kernel. Rest of the paper is organized as follows. Section 2 introduces Dataset, Cosine Similarity, SVM, and NB. Section 3 introduces Result and Section 4 introduces Conclusion.

II. GENERIC BIOMEDICAL EXPERIMENT METADATA MODELING AND MANAGEMENT

A. Dataset

Kaggle provided two datasets, namely tmdb 5000 movies and tmdb 5000 movies. Both csv files namely movies and credits [9] were used with each file containing 20 and 4 characteristics, respectively. Both datasets have been used for Movie Recommendation system. Movies dataset consists of features namely budget, genre, homepage, id, keywords, original language, original title, overview, popularity, production company, production countries, release date, revenue, runtime, spoken language, status, tagline, title, vote average and vote count. Credits dataset consists of features namely movie id, title, cast and crew. The two datasets used in movie recommendation are merged to form a single dataset shown in Figure 1. The columns kept under it include the movie ID, title, genre and tags.

```
                       genres  movie_id                    title  \
0      [action, crime, drama]     49026  The Dark Knight Rises
1  [adventure, drama, action]       254                King Kong
2  [drama, romance, thriller]       597                  Titanic
3     [action, drama, horror]     72190              World War Z
4            [drama, romance]     64682         The Great Gatsby

                                                tags
0  dccomics crimefighter terrorist christianbale ...
1  filmbusiness screenplay showbusiness naomiwatt...
2  shipwreck iceberg ship katewinslet leonardodic...
3  dystopia apocalypse zombie bradpitt mireilleen...
4  basedonnovel infidelity obsession leonardodica...
```

Figure 1 Merged Dataset

B. Cosine Similarity

Cosine similarity [10, 11] is a metric commonly used to evaluate the similarity between two vectors. It disregards the magnitude of the vectors and focuses on calculating the cosine of the angle between them. In the context of movie recommendation systems, it is used to measure the similarity between users or movies based on ratings or other features. The similarity measure plays a vital role in collaborative filtering techniques, which serve as the basis for numerous recommendation systems. Mathematically, the cosine similarity between two vectors C and D is defined as:

$$\text{Cosine Similarity }(\mathrm{C}, \mathrm{D}) = \frac{\sum_{i=1}^{n} C_i \cdot D_i}{\sqrt{\sum_{i=1}^{n} C_i^2} \cdot \sqrt{\sum_{i=1}^{n} D_i^2}} \qquad (1)$$

where, C ・ D is the dot product of vectors C and D. ‖C‖ and ‖D‖ are the magnitudes (Euclidean norms) of vectors C and D, respectively. Ci and Di are the components of vectors C and D at dimension i.

Cosine similarity is a measure that goes from - 1 to 1. A value of 1 implies that the vectors being compared are equal. The value of 0 implies that the vectors are orthogonal, meaning they have no resemblance. The value of −1 signifies that the vectors are completely opposite in direction. When it comes to movie recommendation systems, cosine similarity can be applied in two primary approaches: user-based and item-based collaborative filtering. The objective of user-based collaborative filtering is to identify individuals with similar preferences. Similarity between users is determined by comparing their rating vectors. For example, if users U_1 and U_2 exhibit similar rating patterns, it is probable that they share similar preferences. The algorithm suggests movies to a user U_1 by considering the preferences of other users who have similar tastes. When it comes to item-based collaborative filtering, the main objective is to identify similarities between movies by analyzing users' ratings. Comparisons are made between the rating vectors of movies. If two movies are found to be similar, it is likely that users who enjoyed one movie will also enjoy the other. This approach proves to be highly beneficial when incorporating new users into the system, as it does not depend on having a vast amount of user data.

C. Support Vector Machine

The Support Vector Machine (SVM) [12, 13] is a robust supervised learning algorithm that was developed by Vladimir Vapnik in the 1990s. It is commonly used for classification tasks and can also be applied to regression problems. The SVM algorithm operates by discerning the hyperplane that efficiently segregates data points belonging to distinct groups. When dealing with linearly separable data, the task at hand is to find a hyperplane that can maximize the margin. The margin is the distance between the hyperplane and the nearest data points from each class, known as support vectors. The support vectors are crucial in defining the precise location and orientation of the hyperplane. In an n dimensional space, the equation of a hyperplane can be expressed as:

$$a \times q + e = 0 \qquad (2)$$

The weight vector is denoted by a, the input features are represented by q, and the bias term is denoted by e. The objective is to optimize the margin, which is defined as the ratio of 2 divided by the magnitude of vector a. This optimization is subject to the condition that the data points be accurately identified:

$$y_i(a \times q_i + e) \geq 1 \tag{3}$$

In situations where it is challenging to achieve a complete separation of classes due to noise or overlapping data points, SVM introduces the concept of a soft margin. This requires the incorporation of slack variables ξ i to the optimization problem, which permits a certain degree of misclassification while also imposing a penalty through a regularization parameter C. The optimization objective changes:

$$min \frac{1}{2}\|a\|^2 + C\sum_{i=1}^{n}(\xi_i) \tag{4}$$

$$s.t \begin{Bmatrix} y_i(a \times q_i + e) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{Bmatrix} \tag{5}$$

The linear kernel is the simplest type of kernel, where the decision boundary is a straight line (or hyperplane in higher dimensions). It is given by

$$K(q_i, q_j) = q_i \times q_j \tag{6}$$

The RBF kernel, also known as the Gaussian kernel, is a popular choice for non-linear data. It is defined as

$$K(q_i, q_j) = exp(-\gamma\left\|q_i - q_j\right\|^2) \tag{7}$$

where γ is a parameter that determines the extent of the kernel’s distribution.

D. Naive Bayes

Naive Bayes [14, 15, 16, 17] is a straightforward yet remarkably powerful probabilistic classifier that relies on Bayes’ theorem. It is especially well-suited for classification tasks in the field of machine learning. Despite its straightforwardness, it excels in a wide range of applications including text classification, spam detection, sentiment analysis, and recommendation systems. Bayes’ theorem is a fundamental concept that forms the basis of Naive Bayes. It allows us to calculate the probability of a hypothesis based on the evidence we observe. Baye’s theorem can be expressed as:

$$P(C|D) = \frac{P(C|D).P(C)}{P(D)} \tag{8}$$

The expression P(C|D) represents the posterior probability of class C given feature D. The expression P(D|C) represents the conditional probability of feature D given class C. The term P(C) refers to the initial probability of class C. The term P(D) refers to the initial probability of feature D. The "naive” component of Naive Bayes stems from the assumption of independence. It presupposes that all characteristics are unrelated to one another, provided the category is known. Here is the streamlined model:

$$P(E|Q_1, Q_2, \ldots\ldots Q_n)\alpha\, P(E) \times \prod_{i=1}^{k} P(Q_i|C) \tag{9}$$

In this context, P(E) represents the initial probability of class E, whereas $P(Q_i|E)$ represents the probability of feature Qi given class E. Three primary categories of Naive Bayes classifiers exist, each designed to handle distinct data types: Gaussian Naive Bayes is designed for continuous data, it assumes that the features adhere to a normal (Gaussian) distribution. It calculates the mean and variance for each feature and class, and then uses these parameters to determine the likelihood. The probability density function for a Gaussian distribution is

$$P(q_i|r) = \frac{1}{\sqrt{2\pi\sigma_r^2}} exp\left(-\frac{(q_i - \mu_r)^2}{2\sigma_r^2}\right) \tag{10}$$

Multinomial Naive Bayes is well-suited for analyzing discrete data, particularly word counts in text classification. The likelihood is modeled using a multinomial distribution, which proves to be highly effective for document classification tasks where the features represent word frequencies. The probability of a feature vector given a class is

$$P(x|y) = \frac{N_y!}{x_1!\,x_2!\ldots..x_n!}\left(\frac{\theta_{y_1}^{x_1}\theta_{y_2}^{x_2}\ldots.\theta_{y_n}^{x_n}}{(\sum_i x_i)!}\right) \tag{11}$$

Bernoulli’s contribution Naive Bayes is specifically designed to handle binary or Boolean features. The assumption is that every feature conforms to a Bernoulli distribution, which means it can be used effectively for tasks such as binary text classification, where features indicate whether words are present or absent. The probability of a feature vector given a class is

$$P(x|y) = \prod_{i:x_i=1} \theta_{yi} \prod_{i:x_i=0} (1 - \theta_{yi}) \tag{12}$$

III. RESULT

Based on input, a movie title is randomly selected from the list of good mood genre dataset or bad mood genre dataset. After selecting a random movie, 10 similar movies is recommended to the user shown in Figure 2.

```
Recommendation for movie: End of Watch
Recommended movie: 835          Street Kings
1903           Harsh Times
201          Gangster Squad
930      Observe and Report
496               Sabotage
1137         The Lucky Ones
1785              Stranded
199                Shooter
332              Prisoners
799          Moonlight Mile
Name: title, dtype: object
```

Figure 2. Recommended movies list

To test the accuracy of Recommendation system, two classification algorithms Naive Bayes (Gaussian NB, Multinomial NB, Bernoulli NB) and Support Vector Machine (SVM) using linear kernal with value of regularization parameter C = 10 and radial basis kernel (rbf) with the value of C = 10 and gamma = 0.05 have been applied on 75% and 80% data as training dataset and the result has been formulated in Table 1.

SVM has the accuracy range from 94.69% to 96.80% and Naive Bayes has the accuracy range from 65.93% to 84.40%,

It can be stated that SVM performed approximately same and is quite consistent for all two-training data while Naive Bayes being the worst.

Comparison of Algorithms

Algorithms	Training Size	
	75%	80%
Gaussian NB	84.40%	81.86%
Multinomial NB	71.99%	70.80%
Bernoulli NB	67.91%	65.93%
SVM (linear)	96.80%	94.69%
SVM (rbf)	96.09%	94.69%

Table 1

IV. CONCLUSION

The objective of the paper was to develop a recommendation system that classifies movies based on the movie name which is accessed by the user. From the given dataset, a movie title is chosen randomly from that genre. Afterwards, the system employs a recommendation algorithm based on cosine similarity to provide ten movies that are comparable to the chosen title. In order to evaluate the precision and efficiency of the recommendation system, several classification algorithms were utilized, including Naïve Bayes (specifically Gaussian NB, Multinomial NB, and Bernoulli NB) and Support Vector Machine (SVM) with both linear and radial basis function (rbf) kernels. The algorithms underwent testing on training datasets that

consisted of 75% and 80% of the entire data. The outcomes of these tests were then gathered and presented in a thorough table 1. The results showed that the SVM method demonstrated satisfactory performance, with accuracy ranging from 94.69% to 96.80%. On the other hand, the Naive Bayes algorithm showed the least accurate results, with accuracy ranging from 65.93% to 84.40%, indicating notable discrepancy. The SVM algorithm's consistent and strong performance on all training datasets indicates that it is the most dependable approach for this recommendation system. The fact that it can consistently achieve accuracy levels ranging from 94.69% to 96.80% demonstrates its resilience and appropriateness for the given task. However, the performance of Naive Bayes suggests that it may not be suitable for this particular application, as it exhibits a broad range of accuracy and lower total scores.

REFERENCE

[1] G. Shani and A. Gunawardana. Evaluating recommendation systems. In Recommender Systems Handbook. Springer, Boston, MA, 2011.

[2] S. Raja Rajeswari, S. Naik, S. Srikant, M. K. Sai Prakash, and P. Uday. Movie recommendation system. In Emerging Research in Computing, Information, Communication and Applications, volume 882 of Advances in Intelligent Systems and Computing. Springer, Singapore, 2019.

[3] C. S. M. Wu, D. Garg, and U. Bhandary. Movie recommendation system using collaborative filtering. In 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), pages 11–15, Beijing, China, 2018.

[4] N. Pradeep, K. K. Rao Mangalore, B. Rajpal, N. Prasad, and R. Shastri. Content based movie recommendation system. International Journal of

Research in Industrial Engineering, 9(4):337–348, 2020.

[5] S. Reddy, S. Nalluri, S. Kunisetti, S. Ashok, and B. Venkatesh. Content-based movie recommendation system using genre correlation. In Smart Intelligent Computing and Applications, volume 105 of Smart Innovation, Systems and Technologies. Springer, Singapore, 2019.

[6] Mahesh Goyani and Neha Chaurasiya. A review of movie recommendation system: Limitations, survey and challenges. ELCVIA. Electronic letters on computer vision and image analysis, 19(3):18–37, 2020.

[7] S. Kumar, K. De, and P. P. Roy. Movie recommendation system using sentiment analysis from microblogging data. IEEE Transactions on Computational Social Systems, 7(4):915–923, Aug. 2020.

[8] T. Zhou, L. Chen, and J. Shen. Movie recommendation system employing the user based cf in cloud computing. In 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), pages 46–50, Guangzhou, China, 2017.

[9] The Movie Database (TMDb) and chuan. Tmdb movie metadata, 2017. https://www.kaggle.com/datasets/tmdb/tmdbmovie-metadata.

[10] H. Wen, G. Ding, C. Liu, and J. Wang. Matrix factorization meets cosine similarity: Addressing sparsity problem in collaborative filtering recommender system. In Web Technologies and Applications. APWeb 2014, volume 8709 of Lecture Notes in Computer Science. Springer, Cham, 2014.

[11] E. Bigdeli and Z. Bahmani. Comparing accuracy of cosine-based similarity and correlation-based similarity algorithms in tourism recommender systems. In 2008 4th IEEE International Conference on Management of Innovation and Technology, pages 469–474, Bangkok, Thailand, 2008.

[12] Craig Saunders, Mark Stitson, Jason Weston, L´eon Bottou, and Bernhard Schoellkopf. Support vector machine, 1998. https://www.microsoft.com/enus/research/publication/support-vector-machine/.

[13] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. Algorithmic learning in a random world. Computing Research Repository (CoRR), 2004.

[14] E. M. K. Reddy, A. Gurrala, V. B. Hasitha, and K. V. R. Kumar. Introduction to naive bayes and a review on its subtypes with applications. In Bayesian reasoning and gaussian processes for machine learning applications, pages 1–14. 2022.

[15] Hajer Kamel, Dhahir Abdulah, and Jamal M. Al-Tuwaijari. Cancer classification using gaussian naive bayes algorithm. In 2019 international engineering conference (IEC). IEEE, 2019.

[16] Qiang Guo. An effective algorithm for improving the performance of naive bayes for text classification. In 2010 Second international conference on computer research and development. IEEE, 2010.

[17] Ahmad Ashari, Iman Paryudi, and A. Min Tjoa. Performance comparison between naive bayes, decision tree and k-nearest neighbor in searching alternative design in an energy simulation tool. International Journal of Advanced Computer Science and Applications (IJACSA), 4(11), 2013.