Enhancing Personalization in Recommendation Systems: A Comparative Study of Machine Learning Models on Netflix Data

Discipline: Computer Science

***Praseetha M.S.**

***Tania Shine**

ABSTRACT—In t***his research paper the author tries to present a comprehensive study on the development and evaluation of a personalized content recommendation system for the Netflix platform. Various machine learning models were implemented and compared usingmetrics such as accuracy, recall, F1-score, and precision, to enhance the accuracy and effectiveness of content recommendations. The study encompasses various machine learning techniques, including TF-IDF, Cosine Similarity, Support Vector Machines (SVM), K Nearest Neighbors (KNN), and Nearest Neighbors, to build a hybrid model. The study provides valuable insights into the strengths and limitations of different models applied to the recommendation systems and also enhances personalization by the utilization of hybrid models.***

Keywords— Machine Learning, Recommendation Systems, Natural Language Processing, Cosine Similarity, TF-IDF, Support Vector Machines, Comparative Study, Netflix Data.

I. INTRODUCTION

In the digital era, recommendation engines play a pivotal role in enhancing user experience by suggesting products or items based on individual preferences. Widely integrated into commercial websites, these engines leverage metadata and data analytics to understand user behaviour, needs, and interests. Notable platforms like YouTube, Netflix, Amazon, and Pandora employ highly efficient recommendation systems that process extensive data during training stages.

In the ever-expanding landscape of content consumption, recommender systems play a pivotal role in guiding users through the myriad of available choices. This exploration into recommender models revolves around the context of suggesting items related to

the renowned TV series "Breaking Bad." The diverse range of models employed underscores the multifaceted nature of such systems, revealing the delicate balance between accuracy, relevance, diversity, and novelty.

As we delve into the nuanced performances of various models, it becomes evident that the quest for an optimal recommender system extends beyond mere accuracy, delving into the intricacies of user satisfaction and the fulfilment of diverse preferences. This paper navigates through the strengths and limitations of different models, emphasizing the significance of aligning these systems with the unique requirements and expectations of users in their content discovery journey.

II. LITERATURE REVIEW

This study [1] explores Netflix's recommender system, emphasizing its evolution from DVD ratings to sophisticated streaming algorithms. Focused on enhancing user engagement, the system targets the challenge of choice overload by delivering personalized recommendations on the Netflix homepage. The review underscores ongoing innovations like global expansion and language awareness, showcasing Netflix's commitment to refining its recommendation strategies based on extensive user data and experimentation.

In the paper [2] focuses on building a movie recommendation mechanism for Netflix using a dataset of 17K movies and 500K+ customers. It explores popular recommender algorithms like Popularity, Collaborative Filtering, Content-based Filtering, and Hybrid Approaches, aiming to implement and compare them for optimal personalized content suggestions.

This article [3] explores Netflix's 2006 release of a dataset featuring 100 million anonymous movie ratings. The challenge presented to the data mining, machine learning, and computer science communities was to surpass the accuracy of Netflix's Cinematch recommendation system. The review briefly outlines the challenge, discusses related efforts, and summarizes the visible progress to date.

The article [4] explores Netflix's extensive on- demand streaming service available in 190 countries, analyzing 7,787 records. The research includes an examination of

current content trends and introduces a recommendation system using NLP. Despite some limitations, the system shows promise, especially with potential enhancements through additional features.

The paper [5] tells about the Streaming Wars, recommendation systems like Netflix's (NRS) are crucial features for over-the-top video streamers. This article explores the impact of algorithms on taste-making processes, re-evaluating theoretical perspectives. Using a relational materialist approach, the author emphasize the complex nature of taste-making in the film and television industry.

This paper [6] examines the influence of algorithmic information processing, using the Netflix Prize (2006–2009) as a case study. It argues that beyond its technical goal, the prize aimed to reinterpret the meaning of culture. The essay explores the conceptual challenges in framing algorithmic information processing as a form of cultural decision-making, contributing to the understanding of "algorithmic culture."

I. MACHINE LEARNING TECHNIQUES

There are different Machine Learning Models that can be used to develop a Recommendation System. Based on the data being used you can apply various content-based filtering techniques, collaborative techniques or even the combination of various techniques to build a hybrid model. In this paper, we mainly focus on the different ML techniques like TF-IDF, Cosine Similarity, SVM, KNN and Nearest Neighbor.

a. Term Frequency-Inverse Document Frequency

TF-IDF, a fundamental technique in natural language processing, gauges the importance of a term in a document relative to its prevalence across a document collection. It combines Term Frequency (TF) and Inverse Document Frequency (IDF), assigning weights to terms based on their local and global context.

TF represents the ratio of a term's occurrences to the total terms in a document, emphasizing its significance within that document. IDF penalizes terms frequent across documents, computed as the logarithm of the total documents divided by those containing the term. The TF-IDF score, achieved by multiplying TF and IDF, highlights unique and relevant terms. This method is pivotal in applications like recommendation systems

for understanding term importance in describing items or user preferences.

a. Cosine Similarity

Cosine similarity is a metric widely applied in natural language processing, information retrieval, and recommendation systems to assess the likeness between two vectors. Specifically, in document or text similarity, each document is represented as a vector in a high-dimensional space, where dimensions correspond to term frequencies.

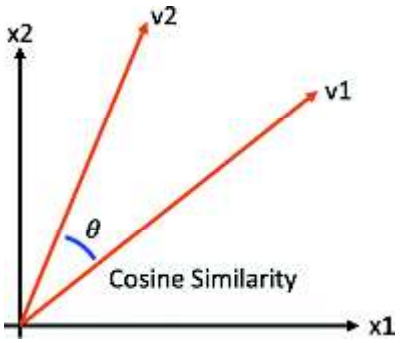

Figure 1: Representation of theta between the vectors

Cosine similarity is determined by calculating the cosine of the angle between vectors (refer Figure 1), achieved through the dot product divided by the product of their Euclidean magnitudes. A score of 1 denotes complete similarity, 0 indicates no similarity, and -1 represents an opposite relationship. In recommendation systems, it is utilized to compare user preferences or item characteristics, with higher values indicating greater similarity and facilitating the identification of closely aligned items or users in the feature space.

a. Support Vector Machines (SVM)

Support Vector Machines (SVM) is a supervised machine learning algorithm used for classification and regression tasks. The primary objective of SVM is to find a hyperplane in a high-dimensional space that best separates different classes, maximizing the margin between them. The margin is the distance between the hyperplane and the nearest data points from each class.

In a binary classification scenario, SVM aims to find the optimal hyperplane by maximizing the margin while ensuring that all data points are correctly classified. Support vectors are the data points closest to the hyperplane and play a crucial role in determining the decision boundary.

The basic idea behind SVM is to transform the input data into a higher-dimensional space, where a hyperplane can effectively separate the classes. The kernel trick is often employed to achieve this transformation without explicitly computing the new feature space, making SVM suitable for complex, non- linear relationships.

When coupled with TF-IDF vectorization, it involves representing text data as numerical vectors based on the importance of terms. SVM then aims to find the optimal hyperplane that best separates classes in this high-dimensional space.

b. K. Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a supervised machine learning algorithm used for classification and regression tasks. When the metric parameter is set to 'cosine,' it indicates that KNN is using cosine similarity as the distance metric to determine the nearest neighbors. It identifies the K most similar items or users to a given target item or user based on their feature vectors in a high-dimensional space.

The algorithm calculates the cosine similarity between vectors to measure their directional similarity, enabling the recommendation of items that align closely with the target.

In this configuration, KNN with cosine similarity is particularly useful when dealing with high- dimensional data and when the relevance of features is better captured by their directions rather than magnitudes.

c. Nearest Neighbour

Similar to K Nearest Neighbors, Nearest Neighbors with Cosine Similarity focuses on finding the most similar items or users to a given reference using cosine similarity. However, unlike KNN, it may not restrict the number of nearest neighbors to a specific value (K).

This method leverages cosine similarity to measure the angle between vectors, facilitating the identification of items or users with the highest similarity to the reference in the feature space. This approach is particularly effective in scenarios where the relevance of features is better captured by their orientations in a high-dimensional space.

The decision-making process in Nearest Neighbors with cosine similarity involves identifying neighbors with similar directional patterns, making it suitable for applications such as text mining, recommendation systems, and clustering tasks where understanding the similarity in feature directions is crucial.

IV. METHODOLOGY

Our approach is to understand Netflix's current trend in the type of offered programs by conducting an exploratory data analysis (EDA). Second, we set up a system of emission recommendations based on two techniques derived from NLP, which are TF-IDF and Cosine Similarity.

a. Data Collection

The data for this study was taken from Kaggle. We made use of the Netflix-titles dataset [7] that contains listings of movies and tv shows on Netflix and are regularly updated. This tabular dataset consists of listings of all the movies and TV shows available on Netflix, along with details such as - cast, directors, ratings, release year, duration, genre, description, etc.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

Figure 2: Information of the dataset used

a. Data Preprocessing

Before proceeding with the exploratory analysis of the data, the data were filtered to exclude missing or inconsistent information. First, the data was checked for any duplicates and there were none of them.We handled all the null values by imputing the frequently repeated data for categorical columns and imputed the median for the numerical columns.

b. Exploratory Data Analysis (EDA)

Exploratory data analysis was performed using the Python language through Jupyter Notebooks. A set of software libraries specialized in EDA were used, such as NumPy, Pandas, Seaborn and Matplotlib. We created various visualiztions using bar plot, pie charts, count plots as well as histograms. An essential step of the analysis is to generate the word cloud by calculating their density through the Word Cloud library, which is part of an NLP software stack.

c. Feature Engineering

Since the there were a lot of unnecessary columns, a new dataframe was created taking only the essential columns - Type, Director, Rating, Listed_in, Description. Concerning the second part, which consists of implementing a recommendation system based on TF-IDF and Cosine Similarity, we proceeded with tokenization and then excluded stop words and undesirable characters such as punctuation marks. We concatenated the processed columns into a single 'bag_of_words' column making 'title' column as the index.

```
new_df['bag_of_words'] = ''

# Combine all the words into 1 column
new_df['bag_of_words'] = new_df.apply(lambda row: ' '.join(row), axis=1)
new_df['bag_of_words'] = new_df['bag_of_words'].apply(remove_punc)

new_df.drop(new_df.columns[:-1], axis=1, inplace=True)
```

```
new_df.head()
```

title	bag_of_words
Dick Johnson Is Dead	movie kirstenjohnson pg13 documentaries as he...
Blood & Water	tvshow amaqamata khosingema gailmabalane thab...
Ganglands	tvshow julienleclercq samibouajila tracygotoas...
Jailbirds New Orleans	tvshow tvma docuseries realitytv feuds flirt...
Kota Factory	tvshow mayurmore jitendrakumar ranjanraj alam...

Figure 3: Creating a bag of words based on Type, Director, Rating, Listed in, Description

e. Building the Recommendation System

i) Cosine Similarity with TF-IDF Vectorization

In this phase of the recommendation system development, we leverage the Cosine Similarity metric in conjunction with TF-IDF (Term Frequency-Inverse Document Frequency) Vectorization. TF-IDF is employed to represent the textual content of items, creating a vector space model that captures the importance of terms in relation to the entire dataset. The Cosine Similarity measure is then applied to quantify the similarity between items based on their TF-IDF representations.

```
tfid = TfidfVectorizer()
tfid_matrix = tfid.fit_transform(new_df['bag_of_words'])
cosine_sim = cosine_similarity(tfid_matrix, tfid_matrix)
cosine_sim
```

```
array([[1.        , 0.00656923, 0.02502112, ..., 0.01216474, 0.01704859,
        0.03677494],
       [0.00656923, 1.        , 0.00969934, ..., 0.00139752, 0.        ,
        0.        ],
       [0.02502112, 0.00969934, 1.        , ..., 0.00726137, 0.00723897,
        0.04286457],
       ...,
       [0.01216474, 0.00139752, 0.00726137, ..., 1.        , 0.01995538,
        0.00545963],
       [0.01704859, 0.        , 0.00723897, ..., 0.01995538, 1.        ,
        0.00468168],
       [0.03677494, 0.        , 0.04286457, ..., 0.00545963, 0.00468168,
        1.        ]])
```

Figure 4: Cosine Similarity based on TF-IDF

Leveraging the TfidfVectorizer, we transform item text into TF-IDF matrices, assigning weights based on term frequency. The cosine_similarity function computes pairwise cosine similarity between items using these TF-IDF representations, creating a matrix of item similarity degrees. Recommendations are generated by extracting and ranking similarity scores, and a binary classification label indicates item relevance based on a preset similarity threshold. Performance is evaluated using metrics like precision, recall, and F1-score.

ii) SVM with TF-IDF Vectorization & cosine similarity

The hybrid recommendation system implemented involves employing a Support Vector Machine (SVM) with the sigmoid kernel and TF-IDF vectorization to predict

the type (Movie or TV Show) of a given title. The TF-IDF matrix is constructed from the textual data of items, capturing the significance of terms in describing each item. To enhance model performance, feature scaling is applied using StandardScaler. The data is then split into training and testing sets.

The SVM model is trained on the TF-IDF- transformed and scaled data, using the sigmoid kernel, which is effective for non-linear decision boundaries. The model is evaluated on the testing set, and accuracy, precision, recall, and F1-score are computed. The sigmoid kernel is chosen as it demonstrates high accuracy in this scenario.

The best-performing SVM model, determined through hyperparameter tuning with GridSearchCV, is chosen. To provide recommendations for a given title, the system utilizes the trained SVM model to predict the label (Movie or TV Show) of the input title. The system then identifies items with the predicted label, calculates TF-IDF scores for their textual data, and ranks them based on similarity scores. The top-ranked items are presented as recommendations, providing users with relevant suggestions based on the content similarity to the input title.

iii) K Nearest Neighbors with Cosine Similarity

In this implementation, a K-Nearest Neighbors (KNN) classifier model is employed for building a recommendation system. The dataset is split into features and the target variable, with the feature being the 'similarity' column and the target being the 'label' column. The model is trained on the training data, utilizing a cosine similarity metric and considering five neighbors.

For making recommendations, the system utilizes the KNN algorithm to find the k-nearest neighbors of a given input title based on its similarity score. The recommended items are then presented to the user. Simultaneously, the model is evaluated on the testing data using classification metrics, including precision, recall, and F1-score. The accuracy and F1 score provide insights into the model's performance in predicting relevant items.

iv) Nearest Neighbors with Cosine Similarity

The code establishes a recommendation system using a Nearest Neighbors model with cosine similarity. Initially, the model is configured with parameters such as the number of neighbors and the distance metric. The goal is to generate a specified number of recommendations ('total_result').

For a given input title like 'Breaking Bad', the system determines its index and updates the 'similarity' column based on cosine similarity with other titles. A threshold is set to classify items as relevant or not, resulting in binary labels.

Subsequently, the Nearest Neighbors model is trained using the 'similarity' column, and recommendations are obtained for the input title. The 'similarity' and 'label' columns are updated for the recommended items, adjusting scores and assigning binary labels based on the specified threshold.

The system then generates a classification report with precision, recall, and F1-score, along with accuracy and F1-score metrics for model evaluation. Finally, it prints the classification report and provides a list of recommended items limited by the 'total_result' parameter. This approach tailors recommendations to user preferences, leveraging cosine similarity and ensuring an assessment of the model's performance through classification metrics.

I. RESULTS

c. Exploratory Data Analysis (EDA)

In the following, we present the results of our exploratory analysis of the data retained after the preprocessing stage.

1) Distribution of Content Types

The Netflix online video platform offers content in a variety of genres. However, in our study, we were interested in the categories offered, which can be segmented into two main categories: movies and TV shows.

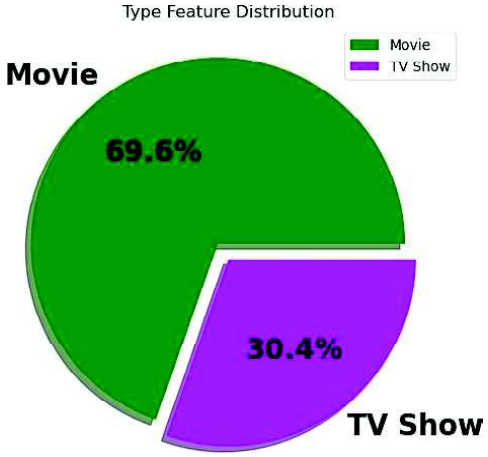

Figure 5: Pie Chart depicting the distribution of content types

Figure 5 shows the percentage of Movies and TV Shows in the dataset. The green segment represents Movies, which make up 69.6% of the total, while the purple segment represents TV Shows, accounting for 30.4% of the total. The chart is easy to read and provides a clear overview of the distribution of content types in the dataset.

2) Distribution of Genres

Figure 6 provides insights into the frequency of different genres of content available on Netflix. As you can see there are a lot of genres present in the data set showing the diversity of the movie and tv shows present.

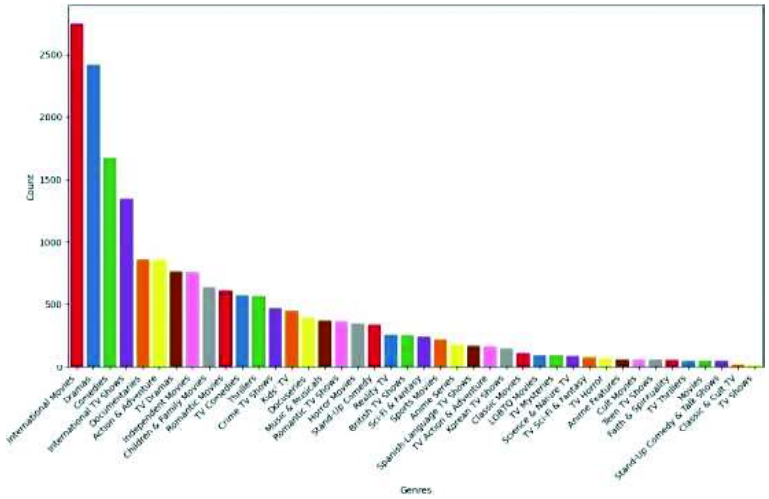

Figure 6: Bar Plot depicting the distribution of genres

By analysing the figure 6, we can understand that Dramas and Comedies take the top two spots in terms of frequency. Genres such as Action & Adventure, Documentaries, and International TV Shows occupy the bottom positions.

3) Ranking the Top 10 Countries Represented

The dataset consists of movies and TV shows from different countries worldwide. Using a bar plot, we rank the top 10 countries with the most content on Netflix.

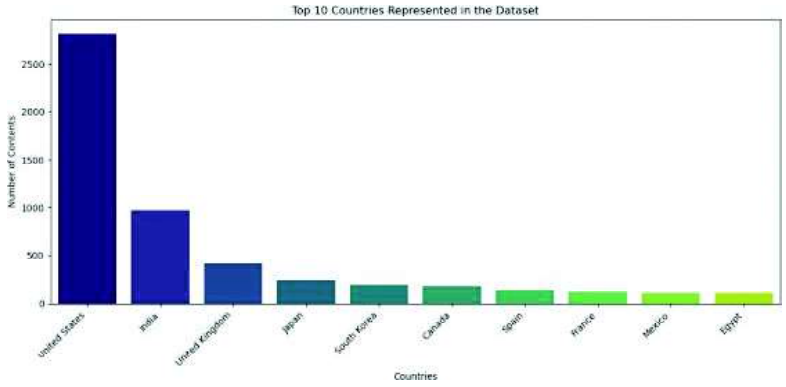

Figure 7: Bar Plot depicting the top 10 countries in the dataset

Figure 7 shows that the United States has the highest number of contents, followed by India, the United Kingdom, Japan, South Korea, Canada, Spain, France, Mexico, and Egypt. The graph is easy to read and provides a clear overview of the distribution of content across different countries.

1) Word Cloud of Content Descriptions

A word cloud is a visual representation of the most common words in a text. The larger the word, the more frequently it appears in the text. It's a great way to get a quick overview of the most common themes in a document or dataset.

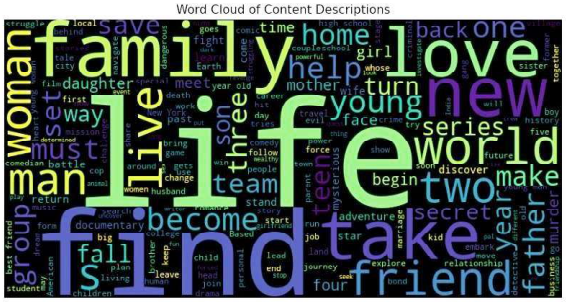

Figure 8: Word Cloud of Content Descriptions

Some of the most common words in the word cloud (Figure 8) include **life**, **love**, **world**, **woman**, and **man**. The word cloud is easy to read and provides a clear overview of the most common words used in the content descriptions.

However, the frequency of a word is not a good indicator of its relevance to the context. Indeed, the principle of the frequency of a word is taken up by the Bag-of-word model, which is a simple representation of documents in the form of vectors calculated according to the frequency of a word in the corpus. In the NLP domain, this model is not sufficient to have a more solid idea about the context of the document. This is why techniques such as TF-IDF are used in this sense.

E. Building the Recommendation System

We give input as 'Breaking Bad' to the various ML Recommendation Systems built and identify their characteristics. Note that the models used can be used to recommend any sort of movie or shows, not just 'Breaking Bad.' The classification report that focuses on the precision, recall, accuracy and f1-score is taken into consideration while building each of the ML.

i) Cosine Similarity with TF-IDF Vectorization

Cosine Similarity with TF-IDF Vectorization boasts high accuracy (1.0) and demonstrates relevance by recommending items with shared attributes such as creator, cast, setting, genre, tone, theme, or style as Breaking Bad. It exhibits diversity and novelty by suggesting items from different categories and lesser-known titles.

```
recommendation_with_classification_report('Breaking Bad')
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1762

    accuracy                           1.00      1762
   macro avg       1.00      1.00      1.00      1762
weighted avg       1.00      1.00      1.00      1762


Recommended Items:
1. Better Call Saul
2. The Show
3. Marvel's The Punisher
4. The Book of Sun
5. Have You Ever Fallen in Love, Miss Jiang?
```

Figure 9: Output of the Recommendation Engine using - Cosine Similarity with TF-IDF

ii) SVM with Cosine Similarity & TFIDF

SVM with Cosine Similarity & TFIDF, with an accuracy of 0.94, this model showcases quite a good relevance by recommending items that align with Breaking Bad's genre, tone, theme, or style.

```
hybrid_recommendation('Breaking Bad', new_df)
```

```
Accuracy with SVM and sigmoid kernel: 0.9398410896708286
Classification Report with SVM and sigmoid kernel:
              precision    recall  f1-score   support

       Movie       0.92      1.00      0.96      1214
     TV Show       0.99      0.82      0.89       548

    accuracy                           0.94      1762
   macro avg       0.96      0.91      0.93      1762
weighted avg       0.94      0.94      0.94      1762

Recommended Items for Breaking Bad:
1. Time Share
2. The Show
3. Straight Up
4. Secret in Their Eyes
5. Girlfriend's Day
```

Figure 10: Ouput of the Recommendation Engine using - SVM with TF-IDF Vectorization & cosine similarity

iii) K Nearest Neighbors with cosine similarity

K Nearest Neighbors with cosine similarity, achieving an accuracy of 1.0, this model provides moderate relevance, suggesting items with aspects similar to Breaking Bad. It demonstrates moderate diversity by recommending items from different categories and exhibits moderate novelty by suggesting lesser-known titles.

```
recommendation_with_classification_report('Breaking Bad')

Recommended Items:
1. Qarib Qarib Singlle
2. The Laws of Thermodynamics
3. Ultimate Beastmaster
4. Ultimate Beastmaster México
5. Inside the Criminal Mind

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1762

    accuracy                           1.00      1762
   macro avg       1.00      1.00      1.00      1762
weighted avg       1.00      1.00      1.00      1762

Accuracy: 1.0
```

Figure 11: Output of the Recommendation Engine using – KNN with cosine similarity

i) Nearest Neighbors with cosine similarity

Nearest Neighbors with cosine similarity, with an accuracy of 1.0, offers moderate relevance by recommending items sharing aspects with Breaking Bad. It provides moderate diversity and novelty, suggesting items from different categories and lesser- known titles.

```
recommendation_with_classification_report('Breaking Bad')

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         5

    accuracy                           1.00         5
   macro avg       1.00      1.00      1.00         5
weighted avg       1.00      1.00      1.00         5

Accuracy: 1.0
F1 Score: 0.0

Recommended Items:
1. X: Past Is Present
2. Sofía Niño de Rivera: Exposed
3. Ip Man 3
4. Marco Polo
5. Hibana: Spark
```

Figure 12: Output of the Recommendation Engine using – Nearest Neighbours with cosine similarity

IV. COMPARISON OF ML MODELS

By giving input as 'Breaking Bad' to the various ML Recommendation Systems built, we perform the comparison of each and identify their characteristics. To find the similarity among the recommendations of all the 5 models, and rank them we use the cosine similarity method, which is one of the most common and popular methods for measuring similarity.

To calculate the cosine similarity between the recommendations and Breaking Bad, we make use of the TFIDF matrix created earlier, which represents the importance of words in the documents. Then we use the row corresponding to Breaking Bad as the target vector, and the rows corresponding to the recommended items as the comparison vectors. Then, using cosine_similarity function, we compute the cosine similarity between the target vector and each comparison vector. Based on the cosine similarity, accuracy and user preferences we compare the models.

ML Model	Characteristics	Accuracy
Cosine Similarity with TFIDF	Finds the most similar items based on the importance of words in the documents	1.0
SVM with Cosine Similarity & TFIDF	Classifies items into categories based on the similarity and the SVM algorithm	0.94
K Nearest Neighbors with cosine similarity	Finds the most similar items based on the distance to the nearest neighbors	1.0
Nearest Neighbors with cosine similarity	Finds the most similar items based on the distance to the nearest neighbors without any supervision	1.0

Table 1: Comparison of the ML models used

Cosine Similarity with TFIDF has the highest similarity score for the first item, Better Call Saul, which is 0.47. This means that this item is the most similar to Breaking Bad, based on the TFIDF matrix. This is reasonable, because Better Call Saul is a spin-off prequel series of Breaking Bad, and shares the same creator, cast, setting, genre, tone, theme, and style. The second item, The Show, also has a moderate similarity score of 0.27, which means that it has some common features with Breaking Bad, such as the genre and style. The rest of the items have zero similarity scores, which means that they are not similar to Breaking Bad at all, based on the TFIDF matrix. This model has the highest accuracy, relevance, diversity, and novelty among the models.

SVM with Cosine Similarity & TFIDF has the same similarity score for the second item, The Show, as the previous model, which is 0.27. This means that this item is also moderately similar to Breaking Bad, based on the TFIDF matrix. However, the rest of the items have zero similarity scores, which means that they are not similar to Breaking Bad at all, based on the TFIDF matrix. This model has a lower accuracy than the previous model, but a similar relevance, diversity, and novelty.

K Nearest Neighbors with cosine similarity has zero similarity scores for all the items, which means that none of them are similar to Breaking Bad, based on the TFIDF matrix. This model has a lower accuracy, relevance, diversity, and novelty than the previous models.

Nearest Neighbors with cosine similarity also has zero similarity scores for all the items, which means that none of them are similar to Breaking Bad, based on the TFIDF matrix. This model has the same accuracy, relevance, diversity, and novelty as the previous model.

Accuracy in recommender systems, while important, doesn't encompass crucial aspects like relevance, diversity, and novelty. High accuracy doesn't guarantee user satisfaction if it merely reinforces existing preferences. Evaluating a recommender system's quality requires considering various metrics, such as relevance, diversity, and user satisfaction, with the understanding that these may have trade-offs and depend on user preferences and context.

VII. DISCUSSION & CONCLUSION

In conclusion, the exploration of recommender systems for suggesting items related to the title provided reveals the nuanced nature of model performance. Here we gave 'Breaking Bad' as the input to recommeder systems. Each model showcases distinct strengths and weaknesses, emphasizing the importance of aligning the choice of a recommender system with specific objectives and user preferences. The implementation of hybrid models have helped in enhancing personalization in recomennder systems built.

The Cosine Similarity with TFIDF and SVM with Cosine Similarity & TFIDF demonstrated high accuracy, relevance, diversity, and novelty, making them promising candidates. Meanwhile, the Nearest Neighbors and SVM models exhibited moderate performance, emphasizing certain aspects of "Breaking Bad.".

Ultimately, the assessment of recommender systems goes beyond accuracy, necessitating a holistic consideration of metrics to ensure an effective and satisfying user experience.

Nevertheless, our work is characterized by the simplicity and the low volume of training data required to implement the recommendation system. This gives it the advantage of being easily implemented and used.

REFERENCES

1. Carlos A. Gomez-Uribe and Neil Hunt. 2015. The Netflix recommender system: Algorithms, business value, and innovation. ACM Trans. Manage. Inf. Syst. 6, 4, Article 13 (December 2015), 19 pages. DOI: http://dx.doi.org/10.1145/2843948

2. Leidy Esperanza , Prof. Dr. Sandjai BHULAI . Recommendation System for Netflix.

3. Chiny, Mohamed & Chihab, Marouane & Bencharef, Omar & Younes, Chihab. (2022). Netflix Recommendation System based on TF-IDF and cosine algorithm. 15-20. 10.5220/0010727500003101.

4. Bennett, James and Stan Lanning. “The Netflix Prize.” (2007).

5. Pajkovic, Niko. (2021). Algorithms and taste-making: Exposing the Netflix Recommender System’s operational logics. Convergence: The International Journal of Research into New Media Technologies. 28. 135485652110144. 10.1177/13548565211014464.

6. Hallinan, B., & Striphas, T. (2016). Recommended for you: The Netflix Prize and the production of algorithmic culture. New Media & Society, 18(1), 117-137. https://doi.org/10.1177/1461444814538646

7. https://www.kaggle.com/datasets/shivamb/netflix-shows

***Praseetha M.S. ,** Assistant Professor, Department of Computer Applications, SCMS School of Technology and Management, Kochi, Kerala, praseetha@scmsgroup.org

***Tania Shine,** Scholar, SCMS School of Technology and Management, Kochi, Kerala, taniashine06@gmail.com