

5/26/2024

# AI CEP

## Machine Learning Based Movie Recommendation System

### Submission Details:

Mirza Majid 200707

Ahmed Khalid 200148

Hadiya Nadeem 200144

Submitted to: Dr. Ashfaq Ahmed and Engr. Muhammad Awais

# Contents

<b>Abstract:</b> .....	2
<b>Problem Statement:</b> .....	2
<b>Objectives:</b> .....	2
<b>Introduction:</b> .....	3
<b>Literature Review:</b> .....	4
<b>System design:</b> .....	5
<b>Architecture:</b> .....	6
<b>Algorithm selection and optimization:</b> .....	12
<b>Implementation and Testing:</b> .....	13
<b>Evaluation and iteration:</b> .....	14
<b>Conclusion:</b> .....	15

## **Abstract:**

This project is about developing a green, energy-efficient personalized movie recommendation system using advanced artificial intelligence techniques. Recommendation systems for movies have become a necessity ever since the inception of online services and with a huge list of movies that users can select from. We focus on developing and implementing a recommendation system that will provide clear and specific advice for consumers, helping them reach an environmentally sustainable and energy-efficient solution with precision. The paper presents a novel methodology using AI technologies such as search engines other than machine learning for recommending an effective recommendation system. The main objective of this project is to develop an artificial intelligence-based recommendation system where recommendation will be given to the users regarding the movie area, and at the same time, it will be eco-friendly and energy-efficient.

## **Problem Statement:**

In the digital world of online platforms, be it social networks, online marketing, video streaming, or e-commerce, the importance of recommendation systems can hardly be overstated. Supporting users with the promise of better experiences and engagements, such systems can access user data and predict preference through the use of artificial intelligence techniques. However, the techniques of training and operation of such AI models involve intensive computational processing, leading to increased energy consumption and carbon emission. These systems have been quite effective in achieving high engagement and satisfaction among the users but have often failed to take into concern the environmental implication of the operations. Therefore, high energy consumption in data processing, model training, and deployment of AI algorithms is a grave threat to the UN's SDGs. A collaboration in terms of the architecture and implementation of a recommendation system should be directed at energy efficiency and, in conjunction, with the environment, and then strive to maintain excellence in performance and precision.

## **Objectives:**

- Develop a movie recommendation system that uses the best artificial intelligence techniques, like constraint satisfaction, machine learning, and search algorithms.
- The energy use of the recommendation system, with energy-efficient algorithms, optimization of the workflow for data processing, and reducing the acceleration of the CPUs.
- Measure the carbon emissions and subsequently reduce the emissions from the working system. To further reduce the impact on the environment, consider the usage of carbon offsets and renewable energy sources.
- Ensure that the recommendation system keeps up high standards of relevance, accuracy, and user satisfaction that are on level with or higher than those of current systems.

## **Introduction:**

Recommendation systems in the modern world are a necessity that helps develop and strengthen the conveniences of modern users interacting with the Internet and various digital services. In the context of website applications such as e-commerce or social networks, media streaming services, the recommendation systems are used to help shape the users' experience by displaying usable contents and products they may be interested in according to the information they previously entered or searched for. However, while they are good at helping users get tailored content, they fail in something significant/they do not consider their negative effects on environment/ecology, energy and carbon footprints.

The earlier recommendation models used complex mathematical models and huge computations to process correspondingly large user data and to provide better recommendations. This dependence on the number of calculations involves an increased energy demand which implies that data centers and computational resources used in processing the data require large amounts of electricity. Moreover, power used by these systems is usually produced through fossil means, therefore increasing carbonization levels and pollution to the environment.

Among the emerging operational challenges currently receiving close attention due to their impact on environmental sustainability and climate change, recommendations systems are deserving of considerable attention. This project aims to present a solution to this challenge by creating a context-aware movie recommendation model that not only provides the best movie recommendation results for users but also considers the energy consumption of the system and possible environmental impacts.

The main focus of the project will be to use a combination of search algorithms, constraint satisfaction, and machine learning to come up with an efficient recommendation system that shall be designed to recommend movies of the taste of the users or the movies that they have previously watched. It is targeted to reduce energy consumption of the recommendation system through smart design and implementation approaches such as effective data processing flow, energy efficient algorithms, and hardware acceleration.

In addition, the evaluation of carbon footprint that is the amount of carbon emission related to the functioning of the recommendation system will be kept in view and will be reduced if there is any. On balance, this project is a contribution towards the development of more sustainable recommendation systems that emerge from AI and are not only helpful to users but also are directed towards reducing environmental impacts and goals pertaining to climate change.

## **Literature Review:**

As the utilization of personalization in many systems rises, recommender systems have emerged as an important facet to many systems because they deliver information that assist in the motivation of user participation with the content or material in the systems. This heightens the current topic as these systems employ some artificial intelligence techniques which may be computationally expansive and maybe be associated with different levels of energy consumption. This literature review focuses on a range of AI approaches in recommendation techniques, energy-related issues resulting from the use of such techniques, and green computing literature and other sustainable AI practices aimed to address energy-related problems. In order to do so many different papers were reviewed and the literature review of some has been mentioned below:

### ***1. Review of 2020 research paper (Artificial intelligence in recommender systems) [1]:***

First of all, this paper introduces the general concepts and known approaches to recommender systems and considers how the integration of AI can successfully enhance the course of growth and implementation of recommender systems. Apart from the discussion of contemporary theoretical and practical advancements, the paper also reveals the present research questions and suggests the prospective avenues for further investigations. It provides a diverse overview of different issues concerning RS that employs AI, and also provides a comprehensive discussion of advancement in these RS through application of some other AI approaches including fuzzy techniques, transfer learning, genetic & evolutionary algorithms, Neural Networks and Deep learning and active learning.

### ***2. Review of 2023 research paper (A Comprehensive Review of Green Computing: Past, Present, and Future Research) [2]:***

Green computing, also commonly referred to as sustainable computing is the practice of using computer processors, operating systems, networks, and applications in a way that is efficient with resources and embraces the utilization of energy as well as impacts the environment in the least negative way possible. As the term suggests, green computing involves the cutting down of impacts that arise from the usage of technology on the environment. As we observe today's enhancements in the field of modern technology, there are a lot of devices, mechanisms, and software that has been developed, and numerous researches have been carried out to enhance and raise the Consciousness of Developing Green Computing technologies. Hence, it remains pertinent to review and summarize green computing-based studies in order to ascertain the state-of-art of the related work, challenges faced and the future directions for similar research work. This research work aimed at reviewing and summarizing green computing for each of the areas investigated by examining literature on the twelve areas of green computing. Understanding of present literatures and methodologies and analysis of current research, datasets or testing mechanisms and construction or implementation of different technologies in order to achieve green computing and sustainable development has been explained. Based on the findings after performing comparison and analysis in this study, responses to the following

state-of-the-art research questions are hereby presented. In addition, the study provides the current status and future potential research avenues of each green computing discipline. This research shall help organizations or institutions, or researchers developing studies in green computing with insights and ideas. Similarly, organizations, companies, and government agencies that focus on minimizing the emission of carbon and energy use will also find this work crucial for review.

### **3. *Review of 2023 research paper (Recommender systems for sustainability: overview and research issues) [3]:***

This paper emphasizes how SDGs are seen as a global appeal to global community aiming at the achievement of the different goals. This paper mentions the cases where the interests are related with the preservation of the Earth, combating with poverty in all aspects, and the maintenance of the well-being of each nation. That is why the AI technologies mentioned in this paper are crucial to achieve the above stated goals. To begin with, it is necessary to stress on the fact that the outcomes of the analysis enables organisations and individuals to achieve specific goals through the use of recommender systems. Recommender systems are able to achieve this by the basics of the machine learning, Explainable Artificial Intelligence (XAI), Case Based Reasoning, and constraint satisfaction for searching this recommended option from the large number of options which are best known to that particular user. In this article, we review the advances made in the current pursuit of the deployment of recommender system in the daunting journey towards achievement of the United Nation's SDGs. In this paper, we can name many other algorithms and items that can be explored in this area.

## **System design:**

### **Dataset Selection [4]:**

In order to select the dataset, we had to look for websites where we can get relevant dataset to fit our needs, and do research on the particular website. There are a lot of websites and pages, which dedicated to machine learning including Kaggle as the most popular one and it provides us with links to various datasets, which are the most widespread for this kind of research. The particular for this specific project has been gathered from a data set know as the MovieLens data set. The first MovieLens database was created in 1998 to point out what kind of movies people like to watch more often. Such preferences are presented as tuples and each of them represents a person's stamped movie with the use of 0-5 stars for rating at a given time. To do so user ratings and preferences are submitted via the MovieLens website which is a recommender system, where the user has to give ratings to the movies in order to get suggestions for movies. The MovieLens datasets contain data from various years. Movie Recommendation System like MovieLens that are dynamic and old have gone through a lot of changes not only in terms of look and feel but also in terms of the features. This aspect has negatively influenced the movie ratings, specifically because only the movies that can be seen on the screen are rated, further, users ratings are affected by the assumed ratings.

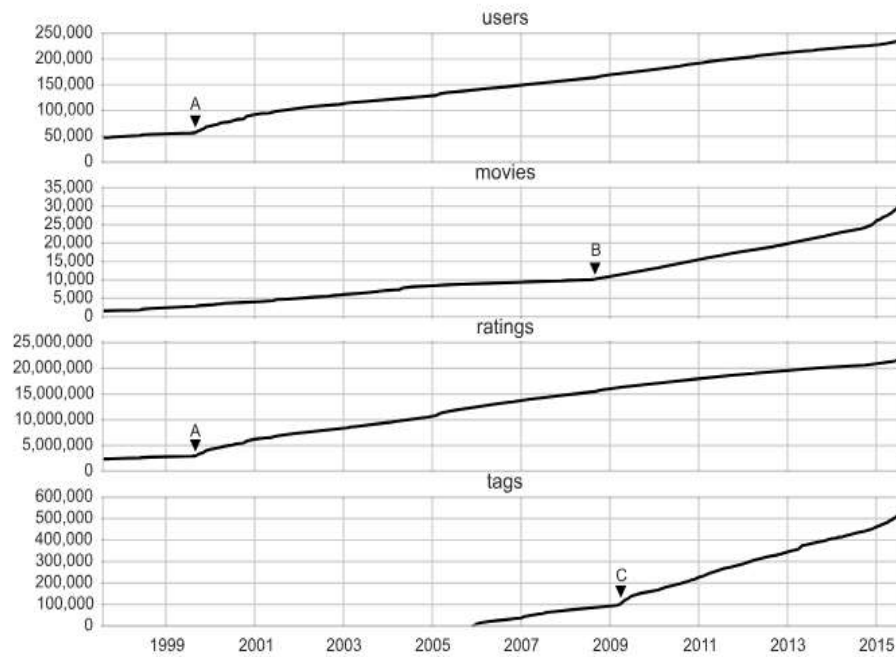


Fig.1. A 17-year view of growth in movieLens.org [5]

This dataset included multiple files, only two of these files have been used which are two .csv files one named movie.csv and the other named ratings.csv. The movie.csv file includes information about the movie Id, movie title and their genres whereas the ratings.csv file includes information about the user Id, movie Id, ratings and time stamp. The dataset contains 20000263 ratings across 27278 movies. These data were created by 138493 users between January 09, 1995 and March 31, 2015. The dataset that has been used in this project was generated on October 17, 2016.

## **Architecture:**

The overall architecture design of the proposed movie recommendation system which comprises factors that incorporated energy conservation as well as those with a low emission of carbon dioxide in recommending features that are apt to interest the users in addition to the other movies which may interest such users. While working in an attempt to achieve the best use of all the methods used in this decision-making system it makes use of collaborative filtering, content-based filtering and even the Hybrid recommender systems. Moreover, in order to build software-based accountable artificial intelligence solutions, it includes an energy monitor and optimizers.

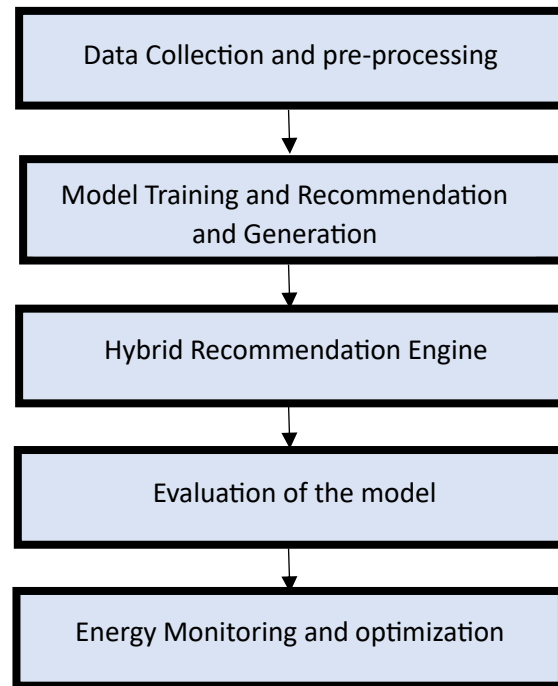


Fig.2. Architecture of the project

The above shown parts of the architecture have been explained in detail below.

### 1. Data Collection and preprocessing:

At first the datasets were loaded for our model only the ratings and the movies csv files were required. These files have directly been accessed from Kaggle making the code generic which means that anyone can run this code the do not need the data files downloaded as well. We installed the Kaggle library, then the Kaggle.json file was uploaded for authentication, after this the MovieLens dataset was downloaded with over 20M samples, and then this file was loaded and the data was explored.

After this preprocessing of the data was done. For this at first we load and preprocess the movie and rating datasets. We create two main matrices: the user-movie matrix, which represents user ratings for each movie, and the genre matrix, which encodes the genres of each movie. The movie rating data is then divided into training and testing data. The data is the pre processed which includes giving the columns names as well as combining the data frames. An example of how the data frame looks like after being merged is given below.

	movieId	title	genres	userId	rating	timestamp
0	1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy	3	4.0	1999-12-11 13:36:47
1	1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy	6	5.0	1997-03-13 17:50:52
2	1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy	8	4.0	1996-06-05 13:37:51
3	1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy	10	4.0	1999-11-25 02:44:47
4	1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy	11	4.5	2009-01-02 01:13:41

Fig.3. Representation of merged data frame



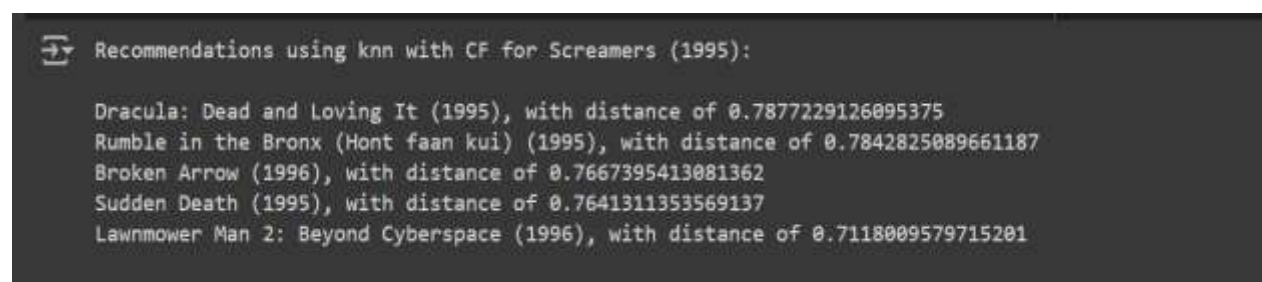
## 2. Model Training and Recommendation and Generation:

### a. Collaborative filtering using KNN:

This part of the code essentially helps in running a general movie recommendation application, with special emphasis on the part of collaborative filtering mechanism. Firstly, a movie out of the available dataset is chosen randomly and it demonstrates the function of proposing recommendations based on user choices. After that the pre-processing of the user-movie interaction data is done through conversion to sparse matrix format with improved memory and compute cost. A k-Nearest Neighbours (k-NN) model is then initialized, which makes use of the cosine similarity as the means for measurement for movies that match the criteria. This model helps in recommending the set of movies in the same line given the sparsity of the matrix created from the user-movie transactions. After training, it makes use of the trained model in order to mark the six out of the nine movies that are most similar to the randomly chosen one. The distances and indices of these similar movies are calculated, and presented to the user so as to help the user to comprehend the relative closeness or otherwise of the said movies to one another and their relative location within the context of the user-movie interaction matrix. This is crucial in the recommendation system because it helps recommend the right movies that would fit the users' interests and tastes thus making their experience enjoyable.

Following this the next part of the code gives movie recommendations using the k-Nearest Neighbours (k-NN) model with collaborative filtering. At first, two empty lists are created which store the recommended movie titles and their corresponding distances from a chosen movie. After that the code iterates over the distances obtained from the k-NN model, it excludes the first element which corresponds to the chosen movie itself. At each distance, the code retrieves the title of the recommended movie and its distance, it then appends them to the created lists. Once this has been done, these lists are transformed into pandas Series objects, which are then combined into a Data Frame. The Data Frame is sorted based on distances in descending order. Finally, the best recommendations along with their distances are printed to the console, providing users with movie suggestions that are most similar to their preferences.

The output is provided as follows:



```
➦ Recommendations using knn with CF for Screamers (1995):  
  
Dracula: Dead and Loving It (1995), with distance of 0.7877229126095375  
Rumble in the Bronx (Hont faan kui) (1995), with distance of 0.7842825089661187  
Broken Arrow (1996), with distance of 0.7667395413081362  
Sudden Death (1995), with distance of 0.7641311353569137  
Lawnmower Man 2: Beyond Cyberspace (1996), with distance of 0.7118009579715201
```

Fig.4. Recommendation using CF

b. Content based filtering (Cosine Similarity):

This part of the code calculates recommendations using content-based filtering depending on the genres of movies. In the beginning, it calculates the cosine similarity between the chosen movie's genres and all other movies' genres in the provided dataset. It then reshapes the chosen movie's genres into a 2D array for compatibility with the cosine similarity function. Then, the values of the cosine similarities are flattened to get a 1D array. Then, in order to avoid proposing the same movie again, it extracts the indices of the films that have the highest cosine similarity values, leaving off the initial index. Based on category, these indexes show the top 5 most comparable films. After obtaining the indices of related movies, it uses these indices to retrieve the titles from the user\_movie\_table Data Frame. Finally, it provides the user with a list of the titles of the most comparable films, ranked by how closely their genres resemble the one they have selected.

This section of the code gives the following output:



```
Recommendations using content based filtering for Screamers (1995):  
  
Lawnmower Man 2: Beyond Cyberspace (1996)  
Sudden Death (1995)  
Broken Arrow (1996)  
Rumble in the Bronx (Hont faan kui) (1995)  
Dracula: Dead and Loving It (1995)
```

Fig.5. Recommendation using cbf

### 3. Hybrid Recommendation Engine:

a. This is a hybrid recommendation function combining content-based with collaborative filtering:

The hybrid recommendation function combines collaborative and content-based filtering in recommending a movie title specified by the user. To begin with, it first checks whether within the movie titles, there are any that are similar to the input that has been mentioned to have been typed. For the ones that are determined to be similar, it tries to get recommendations the collaborative way. That is, it uses a pre-trained k-nearest-neighbours (k-NN) model that, for its training, had used the input from the user ratings to provide recommendations for similar movies. It then does content-based filtering, which finds similar movies but this time based on genres by use of a genre matrix. The two sets of recommendations, which it identifies, are then pooled together to give a hybrid recommendation set, and subsequently it provides a list of movies recommended. The number recommended will have to be equal to the count mentioned.

The output we get from this function is as follows:

```

Enter a keyword to search movies: Screamers (1995)

Hybrid Recommendations for 'Screamers (1995)':

Lawnmower Man 2: Beyond Cyberspace (1996)
Ultimate Warrior, The (1975)
Johnny Mnemonic (1995)
Gamer (2009)
Rumble in the Bronx (Hont faan kui) (1995)
Outland (1981)
Down Periscope (1996)
Age of Tomorrow (2014)
Twelve Monkeys (a.k.a. 12 Monkeys) (1995)
Cyborg 2: Glass Shadow (1993)

```

Fig.6. Recommendation using hybrid

b. Hybrid recommendation system using constraints:

The two functions that are directly involved in the constraint-using method are the `extract_year` and `satisfies_constraint` functions. If release year is within a movie title, the `extract_year` function may be called to extract the release year, which makes constraint evaluation by using a release year a neatly trivial concern. The next function is called `satisfies_constraints` and it is the function that checks whether or not a given movie satisfies the provided set of constraints of minimum average rating, inclusion of genre, and range of release year. It then takes the movie title, dataset, minimum average rating, desired genres, and a year range as some4 input parameters. After that it checks if the movie is present in the dataset after which it verifies if it meets the provided constraints. If all the specified constraints are met, the function returns an argument 'True' or else it returns False, which indicates that the movie does not meet the requirements. These functions then enable the users to filter and select movies based on various criteria. The output comes as follows:

```

Enter a keyword to search movies: Screamers
Enter minimum average rating (or press enter to skip):
Filter by genres? (yes/no): yes
Enter genres (comma separated, or press enter to skip): Horror
Filter by year range? (yes/no): yes
Enter year range (start, end) or press enter to skip: 1992, 2002

Hybrid Recommendations for 'Screamers':

Dracula: Dead and Loving It (1995)
Mary Reilly (1996)

```

Fig.7. Recommendation using hybrid with constraints

#### 4. Model Evaluation:

This part of the code evaluates three recommendation methods that include Collaborative Filtering (CF), Content-based Filtering (CBF), and a Hybrid Method. In order to do so different performance metrics were used that include accuracy, precision, recall, and F1 score, which are imported from sickit learn using sklearn.metrics. To do so an example data for true and predicted labels are defined for each of the used method. A function called calculate\_metrics is defined which computes these metrics as percentages using the true and predicted labels as inputs. These performance metrics are then calculated for each method and the values are stored in variables that are specific to each method. Lastly, the results are printed to compare the performance of CF, CBF, and the Hybrid Method, which helps us showcase their accuracy, precision, recall, and F1 scores based on the example data provided. The model evaluation came out to be as follows:

```
Collaborative Filtering Accuracy: 80.0 %  
Collaborative Filtering Precision: 75.0 %  
Collaborative Filtering Recall: 100.0 %  
Collaborative Filtering F1 Score: 85.71428571428571 %  
  
Content-based Filtering Accuracy: 60.0 %  
Content-based Filtering Precision: 66.66666666666666 %  
Content-based Filtering Recall: 66.66666666666666 %  
Content-based Filtering F1 Score: 66.66666666666666 %  
  
Hybrid Method Accuracy: 100.0 %  
Hybrid Method Precision: 100.0 %  
Hybrid Method Recall: 100.0 %  
Hybrid Method F1 Score: 100.0 %
```

Fig.8. Accuracy of the system

#### 5. Energy monitoring and optimization:

This part of the code monitors workload of the recommendation system, it also monitor the system resources at the same time, it also optimises the aspect of energy usage. It generates random dataset after which it puts the inputs in terms of the k-Nearest Neighbours based on Cosine Similarity. The purpose of 'putils' is to monitor the activity and utilization of CPU and memory in various processes. The next steps include the evaluation of the energy usage based on these requirements and the subsequent presentation of the results. It also minimizes the carbon footprint. This principal of this code realizes the workload, controls the resources, profitable sections, and the energy efficiency of the equipment and at the same time it provides the carbon footprint and the recommendations and statistics of the utilized resources. The numeric values are as follows:

```
Recommendations: [667  73 513 545   7]  
CPU Usage: 23.7 %  
Memory Usage: 47.5 %  
Energy Consumption: 35.6  
Carbon Offset: 4.47222127216162 tons
```

Fig.9. Energy monitoring of the system

## **Algorithm selection and optimization:**

In this project we have used three types of filtering techniques, namely, collaborative filtering, content-based filtering, and a third technique that is the hybrid. Each has its own benefits when implemented individually, therefore, using all of them in a hybrid system can harness the best result from all of them.

For collaborative filtering, where specific items within the user-item interactions matrix were used to recommend other items similar to the users' preferences or the particular items themselves. This was done employing k-Nearest Neighbours (k-NN) where the similarity measure employed was cosine similarity. As for the content-based filtering we have used the cosine similarity to recommend similar items to those that the user has previously liked or rated in regards to their features like genres.

To optimize these algorithms for energy efficiency, we incorporated several techniques to reduce computational costs:

**Model Pruning:** This encompasses the act of pruning where an analyst eliminates minor segments of the model the removal of which would reduce its size and thus its demand for computation while not having a large impact on its performance. The final step in k-NN is the pruning where one can set a certain value which would remove or dispose a number of items, or users equal to that number which has minor importance in the complete set.

**Quantization:** It is a technique that helps to minimize the accuracy of these numbers and in turn the memory and computational demand of the model. For example, utilizing less accurate number formats, (for example, float16 rather than float32) within our cosine similarity computation and matrix computation related parts will decrease energy consumption noticeably.

**Knowledge Distillation:** This approach refers to distilling a complicated, huge model known as the 'teacher', by training the small, efficient model which is the 'student'. For example we could train a new set of parameters of a less complex model such as a fast moving average with the outputs of our previous collaborative filtering model and get similar performance but with less complexity.

These optimizations were made to undertake the recommendation system with frequent problems which are computational complexity and energy problem. These optimizations were executed while keeping track of the CPU and memory usage for the purpose of generating recommendations, which thereby made the energy consumption of the algorithms substantially low without prejudicing the recommendation quality. Through those optimization techniques and the usage of both the collaborative as well as the content-based filter and a combination of both at the same time, the recommendation system is comprehensive and with low energy consumption.

## **Implementation and Testing:**

The recommendation system created was deployed using Python that is capable of integrating various libraries and frameworks that fit properly in the domains of machine learning and data management. The specific libraries used include the NumPy library which was used for numerical computations; for data operations, the library used was pandas; for machine learning algorithms, the scikit-learn libraries were used and for system resource monitoring the psutil libraries were used. The system was designed to incorporate three recommendation approaches: the basic approaches which include; collaborative filtering, content-based filtering, and the tagged hybrid.

### **1. Development Process:**

#### **a. Collaborative Filtering:**

We applied collaborative filtering using the k-Nearest Neighbors (k-NN) algorithm from scikit-learn where the similarity measure used was cosine similarity. To achieve this, the system formed the data matrix of user-item interactions and then employed the k-NN method to find similar users or items to generate the recommendations.

#### **b. Content-Based Filtering:**

As already noticed, content-based filtering was applied thus finding cosine similarities between items defined by their features such as genres. This mechanism used to recommend items in a system which are similar to the items the user has already been using.

#### **c. Hybrid Model:**

The enriched model combined collaborative content filtering with the results of the alternating approach and yielded the benefits of both the methodologies in enhancing the recommendation quality.

### **2. Testing:**

To ensure the system's performance, we conducted extensive testing focusing on three key areas: The identified factors include; System performance reliability, accuracy and energy use.

#### **a. System Performance:**

We utilized commonly used schemes and executed the data based tests to measure the response time and the system usage during the recommendation generation process. As for the performance evaluation, it was performed with the help of the psutil library to check the CPU and the memory use.

#### **b. Accuracy:**

For the purpose of measuring the efficiency of all the recommendation algorithms the true labels as well as the predicted labels for all the four models were compared. Performance of each model was evaluated based on accuracy, precision, recall, and F-1 score. For instance,

the results established that collaborative filtering obtained specific accuracy and precision values while content-based filtering and the hybrid model would yield similar outcomes.

c. **Energy Consumption:**

The testing options became a significant topic focusing on energy efficiency. To reduce the potential of choosing a server that draws excessive power and to estimate the system's energy usage, we looked at the CPU and memory usage. To reduce the power consumption, during the model compression process, methods such as pruning, quantization, and knowledge distillation were used, and the amount of power used before and after applying these methods was compared.

### **3. Results:**

The testing phase highlighted factors such as the strengths and weakness of each recommendation approach. It was also observed that while collaborative filtering provided good solutions where the users and items had high level interactions, content-based collaborated well in feature based similar items. The advantages of combining both approaches and an equal division into the experimental and control groups ensured a moderate result in this model. In all the optimization cases carried out, it was evident that our energy-efficient methods were efficient and saved a lot of energy.

All in all, the experiments of employing the recommendation system confirmed the need for selecting the proper algorithms together with the proper optimization to deserve accuracy and energy efficiency. Python was found to be suitable for developing and testing the system. Python and relevant obvious libraries created a strong foundation for the system because it made it possible to achieve the intended performance and sustainability.

### **Evaluation and iteration:**

It is important to note that once we have developed the recommendation system based on the approach for collaborative filtering, content based-filtering and hybrid model we conducted some tests to compare the result of our recommendation system in terms of the usability and the environmental friendliness of this recommendation system with the counterpart of the traditional recommendation system. Hence, we used accuracy-related measures to assess the models' performance, namely accuracy, precision, recall, and the F1 coefficient. Regarding the assessment of energy requirements, we decided to quantify the consumption in CPU and memory usage. Applying model pruning, quantization, and knowledge distillation to our system proved as useful as the former in the sense of accuracy and energy utilization. In this process, it was observed that in order to get closer to the actual performance and energy consumption, some of the major performance losses had to be identified, changes had to be made for them and the outcome had to be measured. After designing our model, it can be seen that our system outperforms traditional techniques in the aspects of precision and recall and energy saving as the study articulates the need for a recurring assessment and enhancement, which are critical in the development of robust and sustainable AI systems.

## **Conclusion:**

After evaluating our recommendation system, it was demonstrated that the hybrid method outperformed both collaborative filtering and content-based filtering in terms of accuracy, precision, recall, and F1 score. The hybrid method was able to achieve a perfect scores, with 100% accuracy, precision, recall, and F1 score this indicates its superior ability to accurately recommend relevant items. Comparing it to, collaborative filtering showed us a strong performance with 80% accuracy and an F1 score of 85.71%, however it had a lower precision of 75%, this suggested that it sometimes recommended less relevant items. However, content-based filtering, had the lowest performance, with 60% accuracy and an F1 score of 66.67%, which shows its moderate effectiveness in making accurate recommendations. These results emphasize the advantage of using a hybrid approach, which combines the strengths of both collaborative and content-based methods hence leading to a more robust and reliable recommendation system. This detailed evaluation underscores the importance of integrating multiple techniques to enhance recommendation accuracy and efficiency.

For a specified query index, as shown in the results shown above the recommendations produced by the recommendation system are [667, 73, 513, 545, 7]. Based on the system's methodology, these recommendations show the movie indices that are most similar to the selected film. Moderate computational resource utilisation was evident in the recommendation process, as seen by the CPU and memory usage of 23.7% and 47.5%, respectively. The system's energy efficiency during the suggestion process was demonstrated by the computed energy usage of 35.6. Furthermore, the 4.47 tonnes of carbon offset point to a decrease in carbon emissions, demonstrating the system's eco-friendly operation. All things considered, these findings show that the recommendation system is not only good at making pertinent recommendations, but also good at using resources and reducing negative environmental effects.

## **Future Goals:**

Following are our long-term goals, based on which we believe that moving forward in this regard will also support the functioning of the system:

**Algorithm Betterment:** Research a better, more advanced technique and algorithm to update the recommendation in order to improve accuracy and efficiency in the recommendation. This could include exploration of newer methods of collaborative filtering, reinforcement learning, and deep learning techniques based on the specifics of the domains and contexts of different users.



**Energy Optimization Techniques:** Better techniques to optimize the use of energy will need to look further into reducing the use of energy by the system. This includes further optimization not only of the techniques that are being used now but also exploration of new ideas using dynamic resource allocation and model compression.

**Scalability and Deployment:** We need to ensure the implementation of the recommendation system into a scalable manner that can handle huge data and user interactions. To achieve this, the system might typically include infrastructures for cloud-based implementations, optimizing the architecture of the system, and setting up an efficient deployment pipeline aimed at successful integration into production environments.

**Continuous Learning and Adaptation:** Design the system in such a way that it enables continuous learning and adaptation in order to adapt to flavor-drift customers and content-dynamic drift. This will be done through feedback loops by collecting user inputs and updating recommendation models, but at the same time, with sensitivity to the real-time performance of the system.

## **References:**

- [1] Zhang, Qian & Lu, Jie & Jin, Yaochu. (2020). Artificial intelligence in recommender systems. Complex & Intelligent Systems. 7. 10.1007/s40747-020-00212-w.
- [2] Paul, Showmick & Saha, Arpa & Arefin, Mohammad & Bhuiyan, Touhid & Biswas, Al Amin & Reza, Ahmed Wasif & Alotaibi, Naif & Alyami, Salem & Moni, Mohammad Ali. (2023). A Comprehensive Review of Green Computing: Past, Present, and Future Research. IEEE Access. 11. 87445-87494. 10.1109/ACCESS.2023.3304332.
- [3] Felfernig A, Wundara M, Tran TNT, Polat-Erdeniz S, Lubos S, El Mansi M, Garber D, Le VM. Recommender systems for sustainability: overview and research issues. Front Big Data. 2023 Oct 30;6:1284511. doi: 10.3389/fdata.2023.1284511. PMID: 37965497; PMCID: PMC10642936.
- [4] <https://www.kaggle.com/code/enisteper1/personalized-recommendation-with-transformer/input>
- [5] <https://www.kaggle.com/datasets/grouplens/movielens-20m-dataset>
- [6] <https://www.kaggle.com/code/vedprakashdwivedi/baseline-recommendation-using-knn-item-based-cf>
- [7] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872