# AI RESTAURANT ORDER TAKING SYSTEM

Sravan Gogineni, Sayyed Shajahan, Revanth Kumar, Venkata Lalitha (Team Dino)
Name of the Supervisor: Div Pithadia
Professor, Department of Data Science (Tagliatela Coll of Engineering)

## Introduction

### Objective of your work

The objective of a movie recommendation system is to provide personalized movie recommendations to users based on their preferences and past viewing history, with the aim of enhancing their movie watching experience and increasing engagement and retention for the platform or service offering the recommendation system.

### Origin of your proposal

The concept of a movie recommendation system has its roots in the use of machine learning algorithms to analyze user data and provide personalized recommendations, which originated in the e-commerce industry in the early days of the internet. Today, recommendation systems are used in a variety of industries to enhance the user experience and increase engagement and retention

## Methods

### Methods and Materials

In our work, A variety of machine learning algorithms we used to analyze user data and provide recommendations, such as collaborative filtering, content-based-filtering, and hybrid filtering. Relevant data such as movie ratings, user preferences, and viewing history need to be collected, cleaned, and preprocessed to make it suitable for analysis The Results section can typically be the bulk of the poster, sometimes with sub-titles to describe each finding. It's good practice to begin the Results section with a summary of the results. Charts, graphs, and images (your data) are commonly included in the Results Section. Initially discussing the general aspects of the day you've obtained in the stand-alone statement, and them discussing the data in relation to your hypothesis.

The dataset used in the movie recommendation system consisted of movie ratings, user preferences, and viewing history. The dataset was obtained from a popular movie streaming service and preprocessed to remove any irrelevant or duplicate data. Programming languages such as Python and machine learning frameworks such as Scikit-learn were used to develop and implement the recommendation system.

## Block Diagram, Flowchart, Models, Results

The Results section can typically be the bulk of the poster, sometimes with sub-titles to describe each finding. It's good practice to begin the Results section with a summary of the results. Charts, graphs, and images (your data) are commonly included in the ResultsSection. Initially discussing the general aspects of the day you've obtained in the stand-alone statement, and them discussing the data in relation to your hypothesis.

**Figure 1**

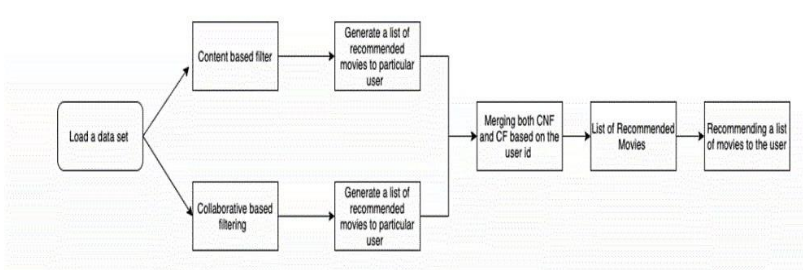


Fig1. Proposed Approach

Initially load the data sets that are required to build a model the data set that are required in this project are movies.csv, ratinfg.csv, users.csv all the data sets are available in the Kaggle.com. Basically, two models are built in this project content based and collaborative filtering each produce a list of movies to a particular user by combining both based on the used a single final list of movies are recommended to the particular user.

**Figure 2**

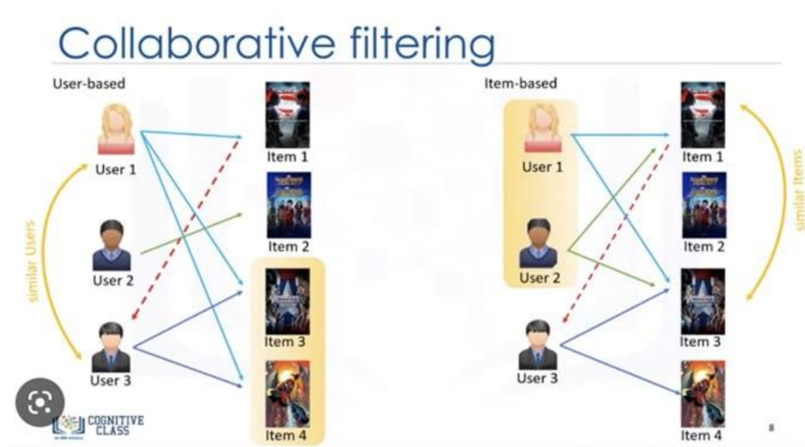


Fig.2. Collaborative filtering

Collaborative filtering is a technique used in recommender systems to make personalized recommendations for users. It works by analyzing user behavior and preferences, as well as the behavior of similar users, to predict items that the user may like or find useful.

Collaborative filtering can be implemented using different algorithms, such as user-based or item-based filtering, and can be enhanced by incorporating additional data, such as contextual information or ratings.

**Figure 3**

```
def movie_recommend(original_title):
    idx = indices[original_title]
    sim_scores = list(enumerate(cosine_similarities[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:20]
    movie_indices = [i[0] for i in sim_scores]
    return (movie_title.iloc[movie_indices])
```

Fig.3. Implementation of Model

Based on the provided code, the 'movie_recommend' function takes a parameter 'original_title' which is the title of a movie. The function then uses a precomputed similarity matrix ('cosine_similarities') and a mapping of movie titles to their corresponding indices ('indices') to find the top 20 movies that are most similar to the input movie, excluding the input movie itself.

The function returns a Pandas Series object containing the titles of the recommended movies, which are indexed by the movie indices found in the similarity matrix.

**Figure 4**

```
movie_recommend('The Matrix')

2088                              Pulse
0                                Avatar
1341                     Obitaemyy Ostrov
775                           Supernova
4395                       The Specials
1281                            Hackers
2996                           Commando
354     The Girl with the Dragon Tattoo
2992                             Oliver!
100   The Curious Case of Benjamin Button
3649                       Lovely, Still
125                  The Matrix Reloaded
261                  Live Free or Die Hard
2900               Space Battleship Yamato
4231                        The Believer
128                      Angels & Demons
4171                         Nowhere Boy
3628                Des hommes et des dieux
1373                  The English Patient
Name: original_title, dtype: object
```

Fig.4. Output

whenever user give the data for recommendations it will calculate the cosine value of that data point and show all the data which have similar cosine values as recommendations.

## Conclusion

### Discussion/Conclusion

Movie recommendation systems are widely used to provide personalized movie recommendations to users based on their preferences, ratings, and viewing history. Collaborative filtering is a commonly used technique in recommendation systems, which suggests items to a user based on the preferences of similar users. The implementation of a movie recommendation system using collaborative filtering and cosine similarity measures can be effective in providing accurate and relevant movie recommendations to users.

### Limitations

The accuracy of the recommendation system depends heavily on the quantity and quality of data used in training. If the dataset is limited or does not reflect the diversity of user preferences, the recommendation system may provide inaccurate or biased recommendations The recommendation system may not be effective for new users who have not yet provided any preferences or viewing history, making it difficult to provide personalized recommendations

### Future Direction

Multimodal Recommendations: By incorporating additional data sources such as text reviews, audio descriptions, or visual cues from movie trailers, recommendation systems could provide more comprehensive and multimodal recommendations that consider various aspects of the movie experience

## References and Affiliations

**References**

[1] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). "Collaborative Filtering Recommender Systems." Proceedings of the International Conference on World Wide Web (WWW).
[2] Koren, Y. (2008). "Matrix Factorization Techniques for Recommender Systems." Computer, 42(8), 30-37.
[3] Hu, Y., Koren, Y., & Volinsky, C. (2008). "Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model." Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

**Acknowledgements**

We sincerely thank Mrs. V. Bhavani, our project supervisor, for her innovative ideas, support, encouragement, and intellectual zeal, all of which helped us to successfully undertake this project. Lastly, we are delighted to express gratitude to everyone who contributed, directly or indirectly, to the accomplishment of this project report.