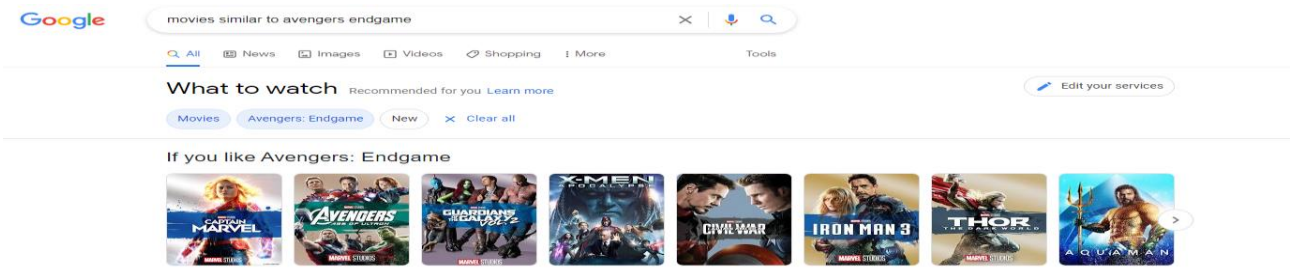
Movie recommendation system

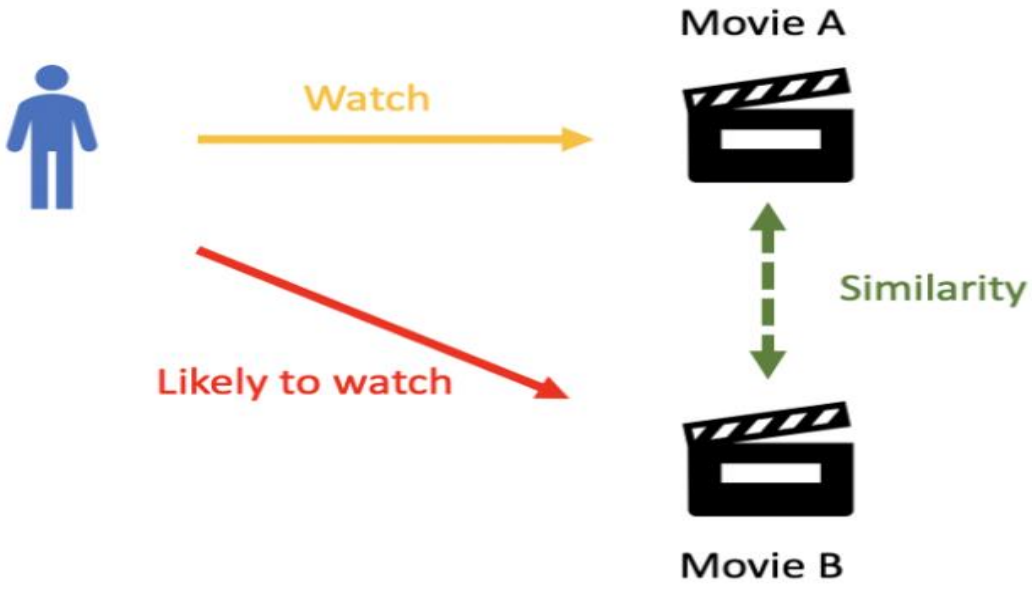
1.Introduction:

We all consume large amount of data in form of entertainment media and movies/ series are important part of it and its vital for us to have a good recommendation system to filter out the data and provide us personalized recommendation as per our taste. Different people prefer different content based on their traits and likings. We all are connected to movies irrespective of our gender, race, age, language, or location but still have our own unique preferences. Some prefer genres like romance, thriller, or sci fi or some like specific actors. Hence, it's difficult to generate a good movie recommendation system for an individual profile. We are creating here a movie recommendation system which will recommend movie suggestions to each individual profile. Every day in our house, we turn on the TV during dinner to watch something and then keep on scrolling the content to find what to select. We want a good movie recommendation system that can suggest us quickly what to watch so that we won't waste time scrolling through the content. Currently, some of the services such as Netflix, Google, Hulu have good movie recommendation system but still struggling with multiple issues such as cold start problem but other countless players like Tubi, Peacock, sling is still struggling to provide a robust recommendation system. In India, people used to use cable service and it's still prevalent for majority of population. Big subscription services are increasing their customer percentage in the country but major old players like Star, Sony are still holding majority of population due to regional content quality and none of them have recommendation systems. With online platforms culture, many small players are entering the market with good content but lack a good recommendation system to compete big player platforms like Netflix. Recommendation system should be robust taking user's content history, ratings and new trending content and not just rely on customer's feedback as the only input as consumer engagement to do can be low considering there is no reward for it or if they are just not happy with the results making these systems perform further poorly. We are suggesting a movie recommendation system that can be used as a service model by any TV providers in entertainment industry that lack one so that any new TV platform entering the market can have a good recommendation system and will investigate current successful recommendation systems such as Netflix, Hulu, and issues they are still struggling and will try to solve some of these problems with emphasis on cold start issue. Our current design of recommendation system can be repurposed for different tasks as per requirements like song, book, clothing, shoes, tourism by using dataset for that service.

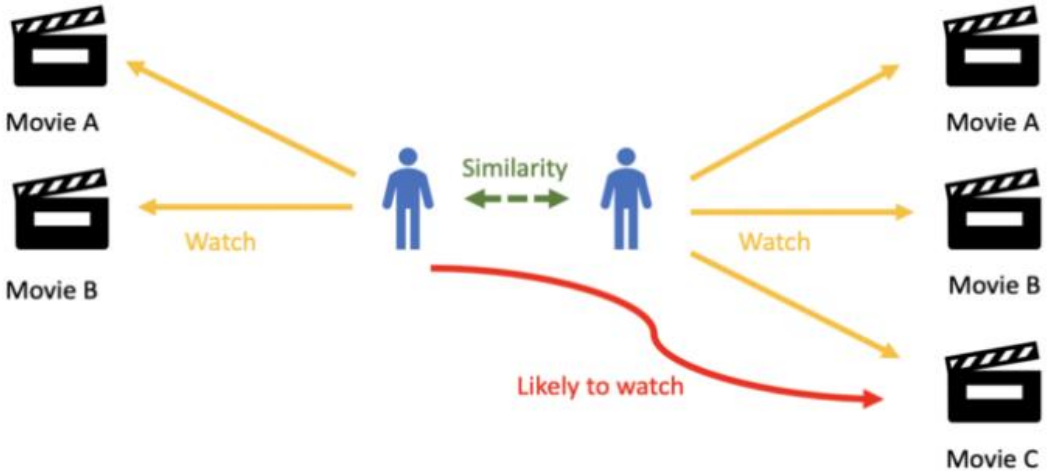


2. Need-finding Analysis and Preliminary Evaluation: Recommendation System main goal is to predict the user's preference towards a particular domain item which is movie in our case and so the focus of recommendation system is to filter and suggest movies/ series to user as per his or her preferences. Mainly, Recommendation systems are based on two types:

Content-based Filtering- Content based filtering recommends products that are like the ones that a user has liked in the past. This similarity (mostly cosine similarity) is computed from the data available for the items (movies) as well as the user's history.



Collaborative Filtering - There are two types of collaborative filtering:



- User-User based Collaborative filtering – Main idea is to find users that have similar preference patterns in the past as the user. Some issues here are people changes their minds frequently or their taste keeps updating. Users are mostly more in number than items and it gets difficult to manage and compute data. Also, fake user profiles can manipulate decision and algorithm can provide false results. Some of the current movie recommendation system based on this suffers same problem such as continuously changing taste.
- Item- Item based Collaborative Filtering – Main idea is to find similar movies instead of similar users and then recommending movies to the users. There are some advantages over user-based CF as movies will not change like people's taste and movies are fewer than people to maintain and compute the similarity.

Movie recommendation systems are also very helpful tool for parents to make sure content watched by kids are age specific and according to survey results most of the parents feel movie ratings are helpful tool in making that decision and ratings are mostly accurate. Survey performed by CARA, Classification and Rating Administration says that 93% of parents said that both the ratings and accompanying descriptors were helpful tools with deciding movies for their kids.[5].

Based on another survey results to decide usefulness of ratings, having binary ratings such as like or dislike are less helpful as compared to user rating from 1 to 5 as it helps in broader classification of content as users can provide range of their likeness to movies. Netflix still switched from star rating to thumbs up/thumbs down which irked their 100 million subscribers and Netfix VP explained that the company gradually realized that behavioral data was a more valuable predictor of the content users wanted to consume. "Users would rate documentaries with 5 stars, and silly movies with just 3 stars, but still watch silly movies more often than those high-rated documentaries." For Netflix, one of the important factors in recommendation is the user behavior data (implicit and explicit feedback) [6].

Hulu recommendation system is based on a collaborative filtering approach. Some of the lessons from Hulu's recommendation system presented in case study [1]:

1. Explicit feedback data is more valuable than implicit feedback data
2. Behaviors study from recent timeline is more important than older timeline
3. Novelty, Diversity, and offline Accuracy are all important factors for design considerations

Because creating movie recommendation system utilizes large amount of data computation, it's still not easy task and many small TV providers are not able to utilize or provide one to their customers. Some providers that provide one are still struggling with accuracy and cold start problem. In India, many online platforms are emerging in the television market with good quality

content, but they all mostly lack a good recommendation system. Having a good recommendation system provide a source of revenue for these services as it promotes customer engagement. Disney partnered with Hotstar in India and organized a hackerearth competition just like what Netflix did in Unites States so that participating teams can provide a good recommendation system to them to compete major players like Netflix India [4]. Hotstar is an on-demand video streaming service in India and have more than 100 million users, so they need a solution that can scale well for its consumers.

Movie recommendation system leveraging user browsing history and content rating and thus personalizing the content for each user creates a value for these streaming services and generates lot of revenue for them. There are some very fundamental issues that are still faced by mostly all movie recommendation systems available currently including cold start problem, data sparsity, and actual usage feedback based on real implementation. Other issues that require further improvements are accuracy and time complexity.

Some important data points to be consider while creating, testing and continuously improve movie recommendation system based on comparison of different movie recommendation systems are:

- Content visited & searched
- Content rated
- Content uploaded & downloaded
- Content finished in single sitting
- Content common features such as genre, actor, director, or language

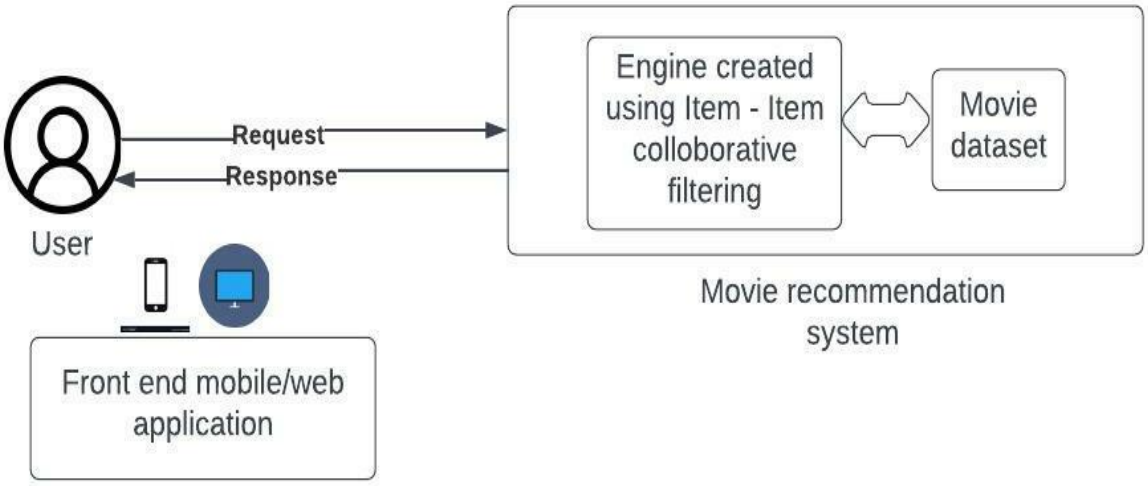
Some of Evaluation measures for a successful movie recommendation system are:

1. User Satisfaction Metrics: This includes user feedback on their satisfaction level in the recommendation system. This metric may have certain issues such as biases, comparison among diverse systems etc.

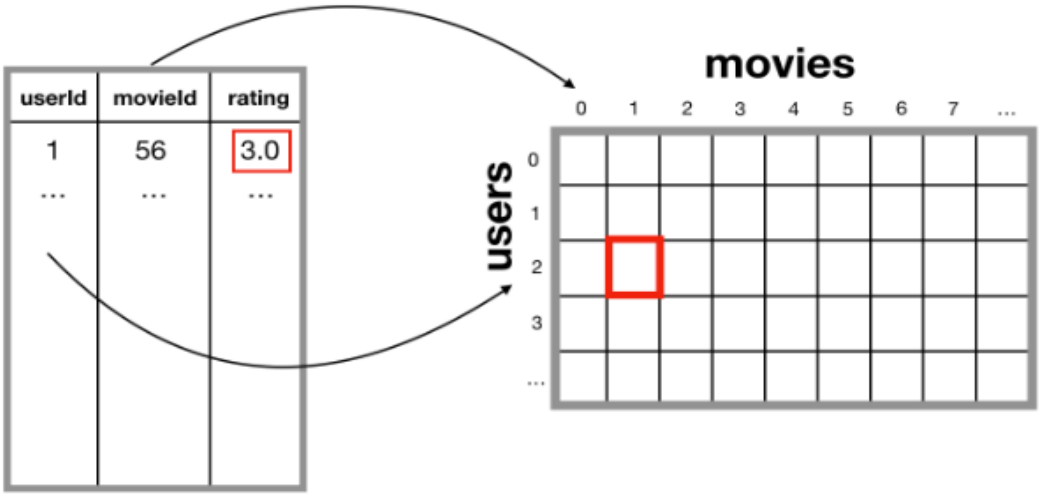
2. Qualitative Metrics: This includes Accuracy

3. Compare recommendation systems on fundamental problems discussed encountered by majority of recommendation systems such as Cold Start Problem.

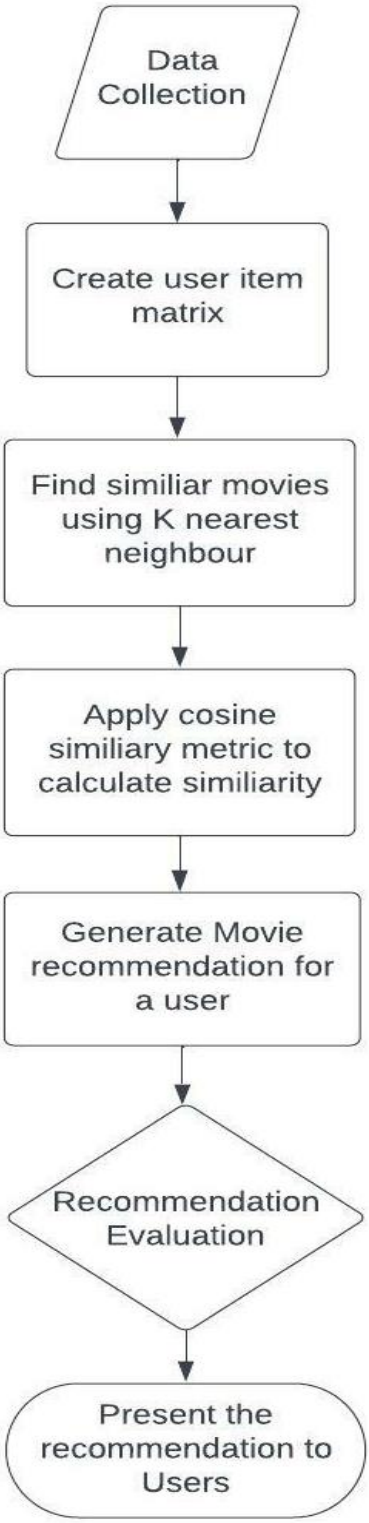
3. Baseline Design Method: We are creating an item-based collaborative filtering movie recommendation system as it provides advantage over user based collaborative filtering based on need finding exercise. We found that its harder to maintain data at user level than at item level. If user likes an item A, and the items B and C are similar to item A in nature, then items B and C are recommended to the user. We have experienced this in Netflix or Hulu where it can be said, "Because you liked movie A, you may also like movie B".



Collaborative filtering is based on unsupervised learning that makes predictions about the user's preferences by learning from interests of larger population. We will create user-item matrix also known as utility matrix where rows represent users and columns represent movies. Collaborative filtering does not need any information about users or features of movies to provide recommendation so it's useful when we don't have detailed feature dataset of movies and in that case this method is preferred.

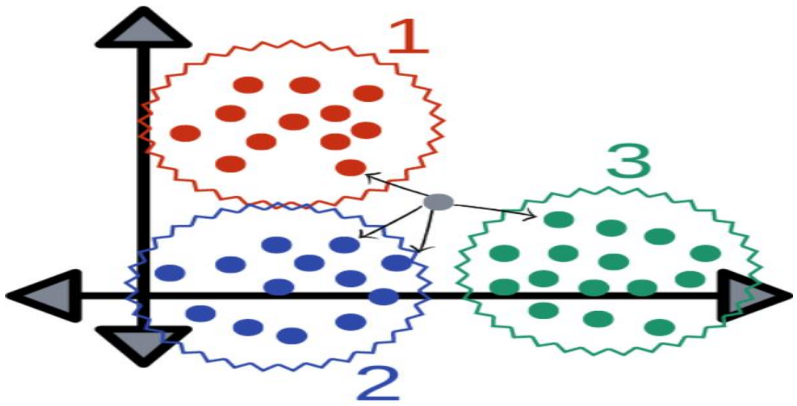


Proposed workflow for calculation of movie recommendation using Collaborative Filtering:



We used pandas DataFrame for representing data and performed data analysis focusing on user provided ratings of movies. Dataset is collected from IMDB dataset. Link of dataset and recommended system code is present in GitHub repository.

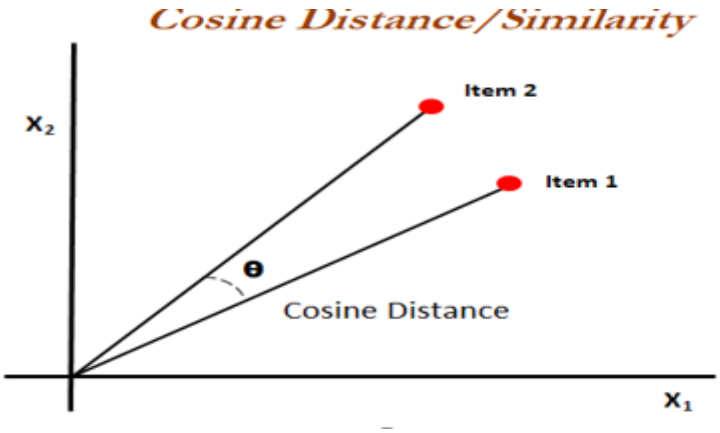
We will be finding similar movies using K nearest neighbor and apply cosine similarity distance metric to calculate similarity score of movies. kNN is a machine learning algorithm to find clusters of similar users based on common movie ratings and make predictions using the average rating of top-k nearest neighbors. Cosine similarity is a method that measures the cosine of the angle between two vectors projected in a multi-dimensional space. The smaller the angle, higher the cosine similarity.



Given two vectors of attributes, A and B, the cosine similarity, $\cos(\theta)$, is represented using a dot product and magnitude as

$$\text{cosine similarity} = S_C(A,B) := \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

where A_i and B_i are components of vector A and B respectively.



Result of movie recommendation is list of movies if you input movie title you have rated in past, and system will generate list of similar movies.

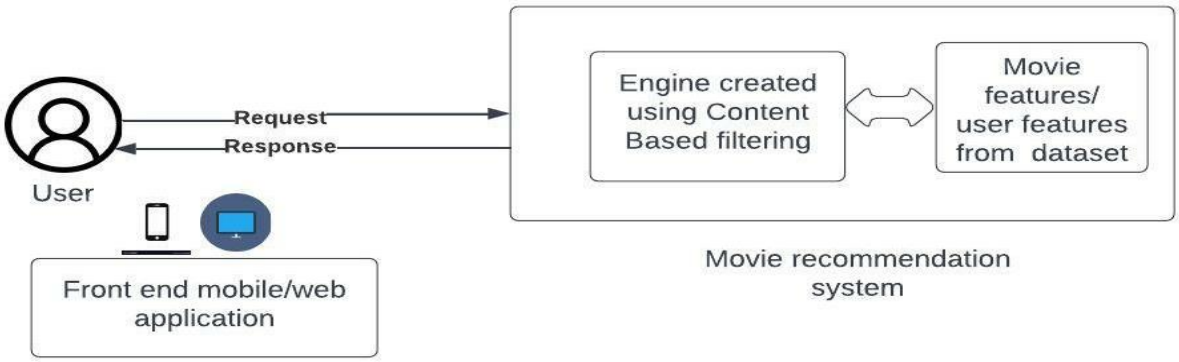
```
Because you watched Toy Story (1995)
Toy Story 2 (1999)
Jurassic Park (1993)
Independence Day (a.k.a. ID4) (1996)
Star Wars: Episode IV - A New Hope (1977)
Forrest Gump (1994)
Lion King, The (1994)
Star Wars: Episode VI - Return of the Jedi (1983)
Mission: Impossible (1996)
Groundhog Day (1993)
Back to the Future (1985)
```

Performance evaluation of baseline design: Collaborative filtering looks at movie similarity based on movies rated by users in past and suggest recommendation to users, but it lacks detailed features of movies which are useful when determining suggestion such as genre, actor, director or even language of movie. While working on movie recommendation system and analyzing the final recommendation results of our item based collaborative filtering, we noticed it suffers from Cold start data sparsity and grey sheep issues, cold start issue occur when current system will not work for new user, new movie or for user that does not like to provide ratings. The sparsity problem occurs when there is less data to work with for any user profile and that impacts accuracy of recommendation. In user item matrix, if user has not rated enough movies, then user item matrix will have few movies to work with and make accurate prediction. This could be common issue as some consumers are lazy or not interested to rate the content. Grey sheep problem occurred when a user seems to rate content in an unpredictable pattern, and it is hard to calculate its obvious closest neighbor. This problem occurred because not all users are diligently rate content and just sometimes randomly like or dislike few movies they have watched. We will investigate cold start issue in detail and will evaluate our system for same. If we are using this baseline design, new items/ new movies should not be considered as they would give false results and reduce the accuracy of system. For new user we could do KYC questionnaire at the time of user sign up as a solution of using baseline system. This will help us add this new user to cluster of users like him/her based the responses in the questionnaire. Baseline recommendation system will add convenience to our lives and make it easier for us to select content to watch based on our ratings but will not work for discussed issues.

4. Design Refinement – In Design refinement, we are changing our engine to focus recommendation system on content filtering and calculate similarity of movies based on feature set of movies such as genre, actor, director, language etc. Collaborative filtering relies mainly on user-item interactions within the utility matrix which was user -movie matrix we created in baseline design. Brand new users or items with no interactions were not included in our baseline design. We are trying to solve various issues we noticed in our results evaluation of baseline design

such as cold start, grey sheep, and data sparsity. Content based filtering can handle discussed problems by generating recommendations based on user and movies features instead of user ratings for a movie.

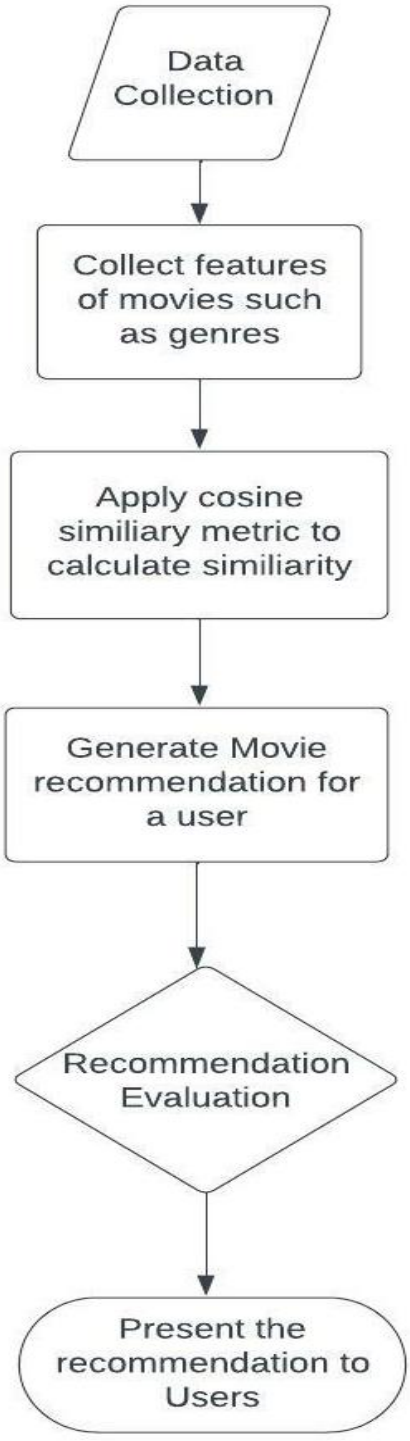
We will also evaluate if we are able to solve cold start issue by using content filtering. This method is meaningful for users that usually avoid providing feedback about content considering they are lazy or busy or not interested in willingly interacting with our recommendation system and we know very little about them. We instead will focus on movie features which is available to us such as genre of a movie it belongs to and year it was released. We can involve multiple features such as actors, directors or language associated with movies to consider international movies as well.



We will load data containing movies features such as genres and user information. We will create similar movies recommendation system by calculating similarity using Cosine Similarity discussed in baseline approach. We started with feature such as genres and can further improve the system by adding further features such as actors, directors etc. Link of code is present in github repository. Final output is result containing movies based on recommendation system when you will input movie name.

```
Recommendations for aladin:
1177                                        Hercules (1997)
95                             Muppet Treasure Island (1996)
673     Land Before Time III: The Time of the Great Gi...
1757                                    Bug's Life, A (1998)
3727                  Ferngully: The Last Rainforest (1992)
6983                                    Rock-A-Doodle (1991)
7758    Asterix in America (a.k.a Asterix Conquers Ame...
8274                    All Dogs Christmas Carol, An (1998)
1390                                          Mulan (1998)
0                                         Toy Story (1995)
Name: title, dtype: object
```

Proposed workflow for calculation of movie recommendation using Content based Filtering:



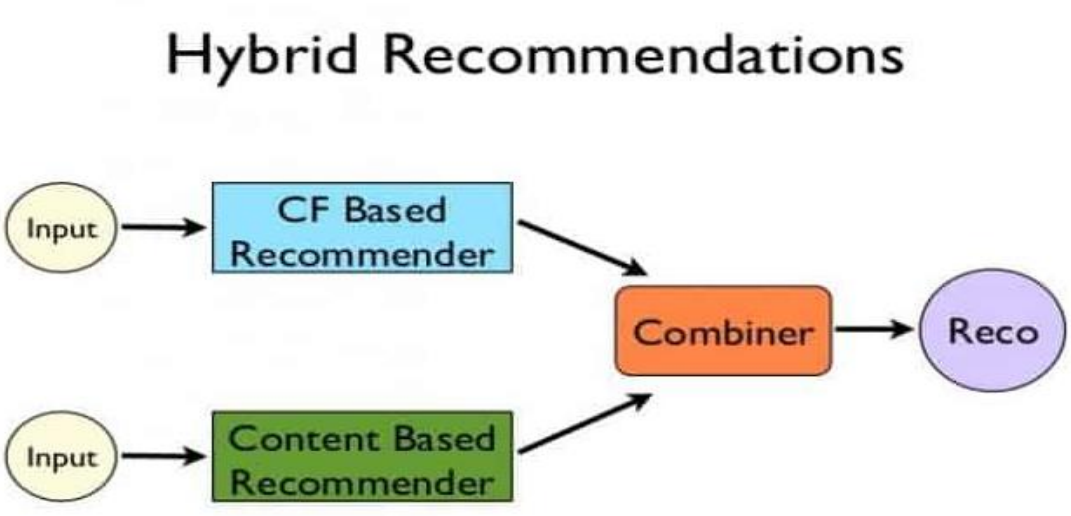
Design can be further enhanced by collecting more data on movies features and create more personalized recommendation by adding meaningful features into data set.

Borderline cases where the proposed design refinement may fail – Design refined approach based on content-based recommendation system can have limitation in certain cases where I might have watched a horror movie because it starred my favorite actor but in general, I do not like horror movies at all. In this case, system might recommend other horror movies to me. System is not very good at capturing complex behaviors or capturing inter dependencies and hence details like these cannot be captured by recommendation system.

Performance evaluation of design refinement approach: We will test our refined approach for encountered issues in baseline design to make sure it solves issues encountered in baseline approach such as cold start, grey sheep, and sparsity and will evaluate system for cold start issue. When we checked accuracy for both baseline system based on collaborative filtering and refined approach based on content-based filtering, we noticed that for users who rate content regularly and we have enough data for them, performance of collaborative system is better than content-based recommendation system. We are suggesting another design refinement of creating a hybrid model to improve accuracy of recommendation system and solve discussed issue.

Design Refinement 2

Create a Hybrid system to improve quality of results. Hybrid system is when you utilize both collaborative and content-based filtering for creating a recommendation system. Hybrid system follow a blended approach that combines two techniques and attempts to use the advantages of one to fix the disadvantages of other. Design of recommendation system can be further enhanced by this approach. Hybrid system is designed by approach of using CF based system if we have enough user rating data. So, for active users who engage themselves with recommendation system providing ratings on movie frequently will fall on CF based system but users with no active engagement or new users that are just introduced in the system will be considered by content-based recommendation system.



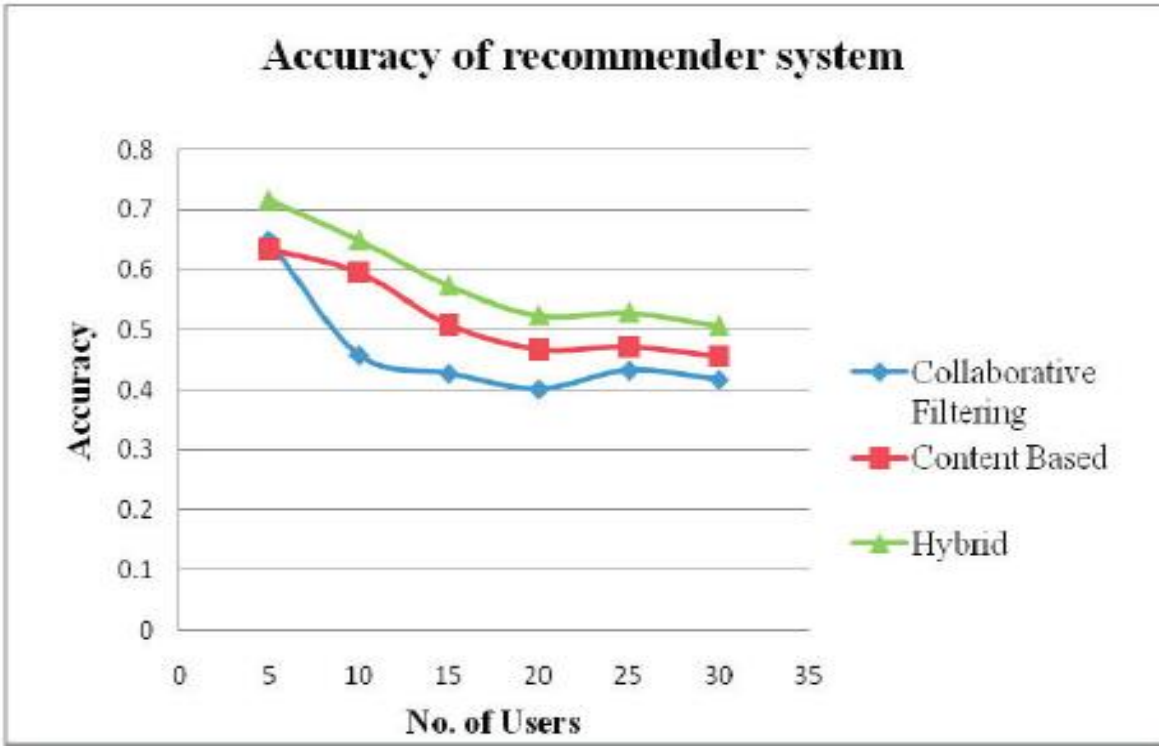
We created Hybrid recommendation system that considers both collaborative and content-based recommendation system. Link of code is present in github repository.

```
Because user_id : 1 liked Toy Story (1995)
Recommending below movies through collaborative filtering
3114            Toy Story 2 (1999)
480            Jurassic Park (1993)
780            Independence Day (a.k.a. ID4) (1996)
260            Star Wars: Episode IV - A New Hope (1977)
356            Forrest Gump (1994)

Because user_id : 2 liked Toy Story (1995)
Recommending below movies through content based filtering
1706              Antz (1998)
2355       Toy Story 2 (1999)
559          Space Jam (1996)
1357    Borrowers, The (1997)
1757     Bug's Life, A (1998)
Name: title, dtype: object
```

Above example demonstrates how the hybrid model performs better overall. We have 2 types of users; the first set give feedback regularly and the other does not or is new to the platform. User 1 regularly rates content and usually likes action and classic movies, but he gave good ratings for Toy Story movie recently. As you can see the recommendations for him reflects not only his recently liked movies but his overall movie preferences. User 2 is new to the platform and hasn't rated many movies. We don't know much about him. The hybrid model detects that we don't know much about this user and only know that he liked Toy Story. As you can see the recommendation reflects that using content-based filtering. The movies recommended to him belong to the same genre and in some cases are a sequel or prequel of the movie the user liked. If we were to get movie recommendations user using collaborative approach, we won't get any recommendation or get poor quality of recommendations.

Performance evaluation of design refinement approach: This system provides best accuracy among first two and solves discussed issues such as cold start for new users or new movies. Hybrid based movie recommendation system works best for wide range of users. This solution is also scalable, and it will break the curse of dimensionality by reducing item/ user matrix space.



Additional Design Refinements for improving current system for future work:

1. Work on improving refined content-based recommendation system by considering extensive data to improve country specific recommendation such as as for India where content is divided between Bollywood, Tollywood, and many regional divisions. Try to work on data on regional movies to cater crowd and create a recommendation system accordingly.

2. Work on *implicit feedback data* of users to improve refined content-based recommendation system. Netflix believes implicit feedback was very useful in their journey of improving their recommendation system. Explicit feedback which we used in our baseline design is direct feedback towards movie such as movie ratings. Implicit feedback refers to indirect behavior towards movie watching such as if person finished movie in one go or left in middle or if person is returning for different seasons of that show or different parts of that movie. Implicit feedback makes assumptions about a user's preference based on their actions. For example, when we love a certain show and binge-watch that show and finish all seasons in a week, there's a high chance that we absolutely like that show but if you leave a series after watching 1st episode, you are not so into it. Implicit feedback data is also very important when you don't like to rate a movie as 4 or 5 but you might still like to watch it for pleasure and fun. We can implement implicit feedback recommendation system using matrix factorization which is particularly useful for very sparse data and can enhance the quality of recommendations. The algorithm works by factorizing the original user-item

matrix into two factor matrices: user-factor matrix (n_users, k) and item-factor matrix (k, n_items).

3. Work on *model based collaborative filtering* method to develop a user model using ratings of each user to evaluate the expected value of unrated movies. We will use machine learning and data mining algorithms to create a model. The model will be developed using the utility matrix which is build using rating given by user for each movie. The model is then trained by getting the information from the utility matrix. Matrix Factorization method can be used where rows represent users and movies as columns. Major advantage of model-based approach is real time prediction based on generated model that provides very high scalability and high-performance speed, but we still need to solve other encountered issues we solved using content filtering. The issue still present in this method is that the prediction of movie for a given user, movie pair is the dot product of the corresponding embeddings. So, if movie is new, the system cannot generally create an embedding for it and hence cannot query the model with that movie known as cold start problem. Model based method also adds inflexibility in updating ratings real time which can impact quality of predictions. Adding sentimental analysis to make model strong is the key. Use sentiment analysis on movie reviews given by user to understand their taste, likes of movies and suggest them similar taste in future to make system robust. Another benefit of this method is improving model by training it on huge variety of data and create a system that can handle complex scenarios and inter dependencies that content-based filtering system may miss.

We can utilize above design refinements and can create a *hybrid model-based recommendation system* that can solve discussed issues such as cold start problem by training the model on enough data and consider ratings as well as features of movies. Model based system with highly trained ML model will provide benefits of improved performance and updating results in real time.

5. Evaluation Plan: Since recommendations in general are so subjective it is very difficult to judge the quality of the recommendations in a quantitative manner. The best way to determine the quality of the recommendation is by asking the user themselves how they feel and if they like the recommendations that were given to them. Below are some of the metrics I plan to calculate to determine the performance of the system.

We will compare baseline and refined approaches on below metrics:

1. Compute time

- Training
- Evaluation/Running

2. Storage requirements

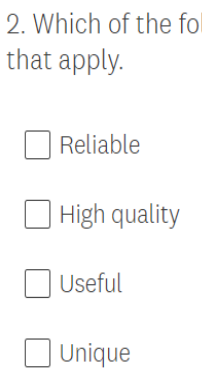
- Data Structure used

- Max data size needs

3. Empirical Evaluation

- How it performs on new movies – Cold Start Problem
- Qualitative Metrics includes Accuracy

4. Qualitative Evaluation - User Satisfaction Metrics: User survey of how effective the recommendation is? Survey is designed with list of questions that are easy to answer and take few seconds to get feedback from users on use case and value of recommendation system. I can send the link to various users of different demographics to get the results and then evaluate the use case and efficiency of system. I have not gathered any personal information in the survey to maintain the privacy. We will try to limit the biases by sending and collecting data from different people from diverse background such as age groups, cultural, race, country, and language to make sure.
Created the survey to get user response on recommended systems -
https://www.surveymonkey.com/r/55XS2BS



6. Evaluation Execution and Results:

1. Compute time

- Training:
    - For N users and M movies, the time complexity for collaborative filtering is $O(M*N^2*K)$ in the worst case where is k is the hyper parameter for kNN. but since only some users rate only some movies the data is sparse and the approximate time complexity is $O((M + N) * K)$.
    - For M user and N features (actor, genre, director) of the movie the time complexity of content based filtering in $O(M*N)$

    - Since hybrid approach uses both the techniques the overall time complexity for this approach is average of the two.
- Evaluation/Running: Generating recommendations for a given user is O(K)

2. Storage requirements

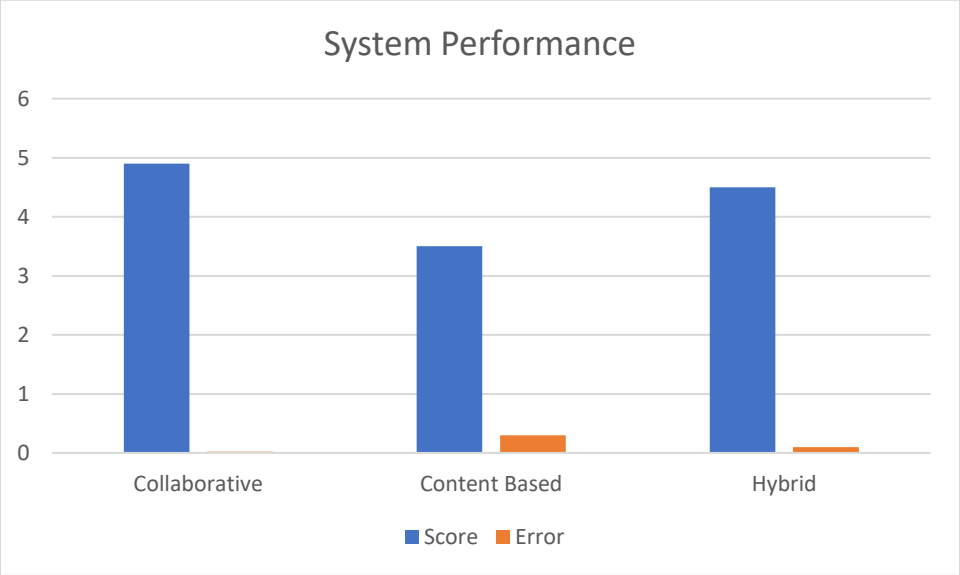
- Data Structure used: Pandas data frame / Matrices
- Max data size needs: O (M * N).
    - The IMDB data needed for solving this problem is around 8GB, it can be loaded in memory to perform computations and hence memory does not cause any concern for this task.

3.Empirical Evaluation

We used the dataset from IMDB to conduct experiments. We fetched the various movie feature dataset such as genre, decade it was released. We created this system for around 10k movies and 10k user ratings on various movies. We randomly deleted 20% of the ratings and used our systems to determine recommendation and then compare it with users actual rating for those recommendations. We randomly selected so that data selection happens across all features of movies like genre, year so that we can reduce bias in our experiment dataset.

- How it performs on new movies – Cold Start Problem
    - We will check all three movie recommendation systems, baseline based on collaborative filtering and refined methods based on content-based filtering and hybrid method for new movie (we can use movies for which we deleted rating in dataset for our experiment) and observe the behavior for both the baseline and refined approaches. Baseline method cannot output any result and content based and hybrid approach will output results.
    - Since we are using hybrid approach to solve this problem, we can see that the system handles both scenarios, user new to the system and movie new to the system very well.
- Qualitative Metrics includes Accuracy
    - On comparing the ratings, the given by user to the recommended movies we see that the rating is highest when the hybrid system can run collaborative filtering. It is decent when the hybrid system is using content-based filtering.

- Below is the bar graph denoting the performance of the system when we randomly deleted 20% of the ratings and used our model to determine recommendation and then compare it with users actual rating for those recommendations.



4. **Qualitative Evaluation:** Qualitative evaluation requires users to submit survey which can be sent periodically to them to get feedback on the performance of system. This metric is the most important metric to judge the performance of the system. But the problem with this approach is it is very difficult to get users to fill up a survey form without incentive. And it would be difficult to determine the quality of the responses if the survey if the users are incentivized. I have not got enough response on my survey as system was in progress when I designed survey. Data analysis can be performed further on system performance based on survey results.

Conclusion

In this paper, we started a baseline design of movie recommendation system based on item item collaborative filtering and noticed that system was suffering from some issues such as cold start problem and is also not suitable when user is not actively engaged in providing ratings for movies. We noticed it works best for users who are actively providing ratings and if you have good amount of data on user ratings for movies, this recommendation system works best for such users. We then worked on design refinement and created content-based filtering recommendation system to solve various problems such as cold start and data sparsity. When we evaluated this system, we noticed that it solves cold start issue as this considers feature sets of movies such as genres instead of relying on user ratings. This system has various advantages over collaborative as it considers wide range of feature sets of movies and recommend movies to users without looking at its historical ratings or data. Content based filtering also provides user privacy as no historical data is being considered. We then compared accuracy of these two systems and collaborative filtering had edge over content based for users that were actively providing ratings for movies in the system, but content based worked best for a large dataset where users were not engaged, or new movie or user were added to system. Finally, based on various evaluation and data analysis, we proposed a Hybrid based movie recommendation system which combines collaborative filtering and content-based filtering to solve cold start problem for new movies and improve overall accuracy of system for all different variety of users. Hybrid approach works best as it takes benefits of both collaborative and content-based filtering system. Hybrid system also provides scalability as system is created using clustering to reduce the dimensionality of the data. Overall, we improved the performance of system using Hybrid approach by increasing overall accuracy of system and solved cold start and data sparsity issue. We suggested model-based hybrid approach for future work to create neural nets based on deep learning to further enhance performance and achieve better scalability. Overall, our goal was to survey different approaches and can create a recommendation system that can be utilized by new providers entering the market that lack one. Recommendation system is a necessity in current world for different entity and we suggest that current system can be repurpose for other recommendations such as books, songs based on use case.

References:

1. Leban, J. (2020, May 21). *Essentials of recommendation engines: Content-based and collaborative filtering*. Medium. Retrieved April 24, 2022, from https://towardsdatascience.com/essentials-of-recommendation-engines-content-based-and-collaborative-filtering-31521c964922
2. Srikanth, P. (2017, November 23). *3rd place winning solution for building a movie recommendation engine for Hotstar*. Medium. Retrieved April 24, 2022, from https://medium.com/data-science-analytics/building-a-movie-recommendation-engine-for-hotstar-478fb4b21c17
3. Graves, J. (2018, September 5). *Survey shows 93% of parents find film ratings helpful in making movie choices*. Motion Picture Association. Retrieved April 24, 2022, from https://www.motionpictures.org/press/cara/
4. Balboni, K. (n.d.). *5 stars vs. Thumbs up/down-which rating system is right for your app?: Appcues blog*. RSS. Retrieved April 24, 2022, from https://www.appcues.com/blog/rating-system-ux-star-thumbs
5. Christakou, C., & Stafylopatis, A. (2005). A hybrid movie recommender system based on Neural Networks. *5th International Conference on Intelligent Systems Design and Applications (ISDA'05)*. https://doi.org/10.1109/isda.2005.9