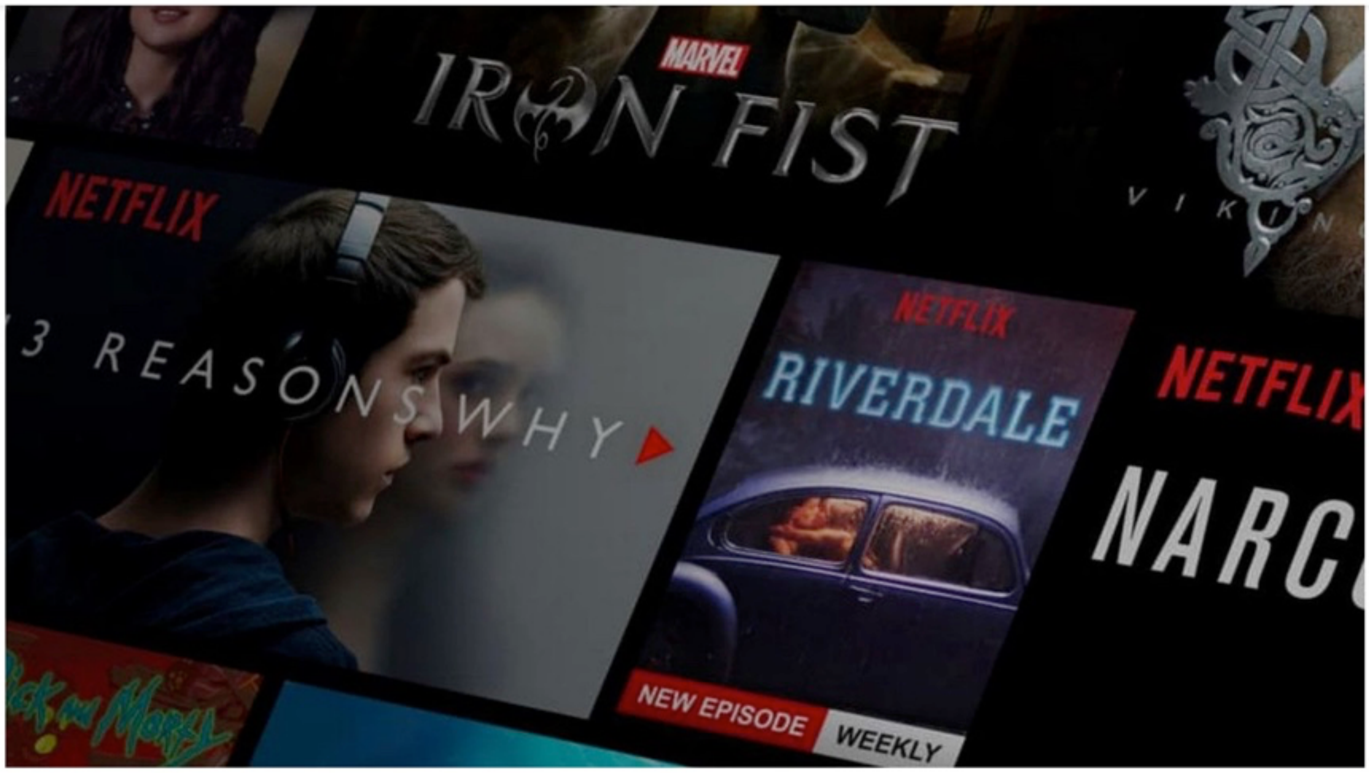
Content and Collaborative based movie recommender system with sentiment analysis



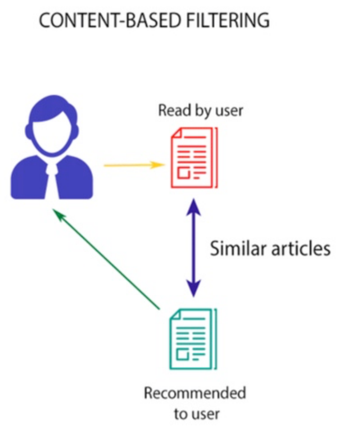
Problem: why recommender system needed:

Recommender systems help the users to get personalized recommendations, helps users to take correct decisions in their online transactions, increase sales and redefine the users web browsing experience, retain the customers, enhance their shopping experience. Information overload problem is solved by search engines, but they do not provide personalization of data, it helps users discover items they may like

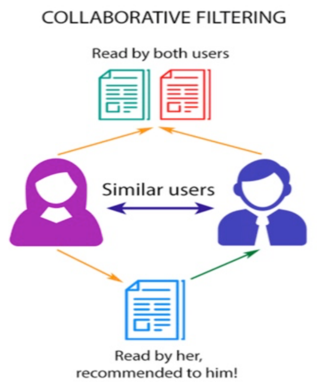
Types of recommender system

1. Content-based filtering
2. Collaborative filtering
3. Hybrid model

1. **Content Based** Movie system recommends movie on the basis of the content like the content description/synopsis, cast, crew, director and genres of the movie.



2. **Collaborative based** movie system recommends movie based on
   a. ratings by similar users and predicting movie rating from past user ratings



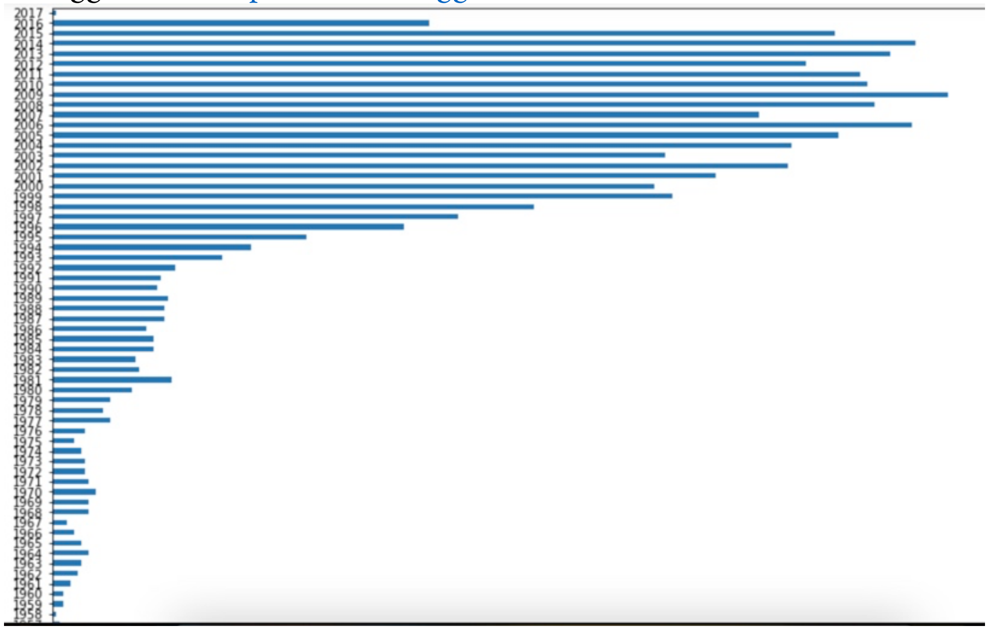
3. **Hybrid model** : can be considered as the combination of the content and collaborative filtering method

1. Content-based movie recommender system : Content-based filtering makes recommendations by using keywords and attributes assigned to objects in a database (e.g., items in an online marketplace) and matching them to a user profile. The user profile is created based on data derived from a user's actions, such as purchases, ratings (likes and dislikes), downloads, items searched for on a website and/or placed in a cart, and clicks on product links

Steps followed:

1. **Data Acquisition**

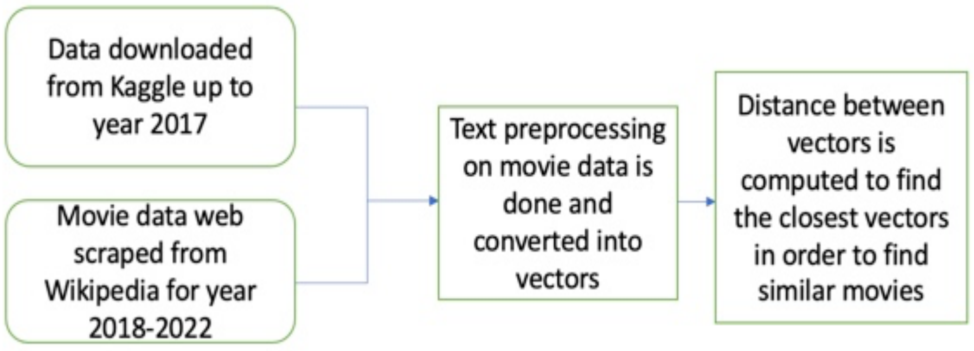
1.Kaggle link: https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata



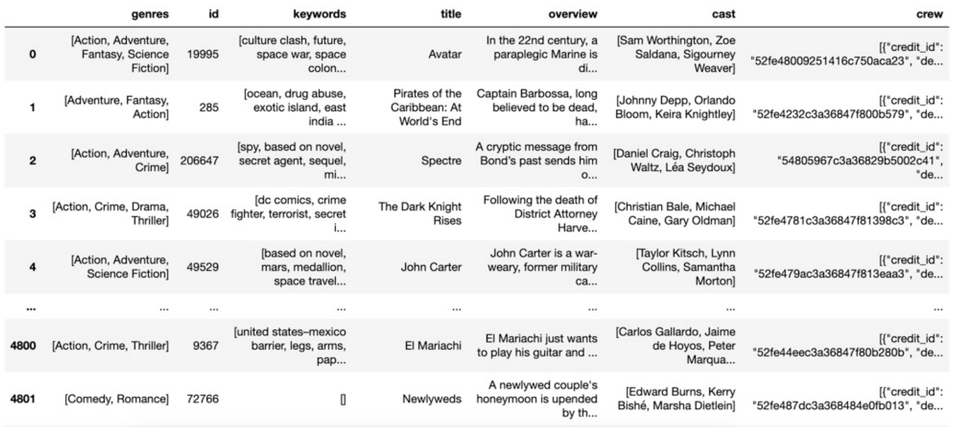
Here data is available only up to year 2017

2. Web scraping is done to get the list of movies from 2018 – 2022 from Wikipedia

Model Architecture:

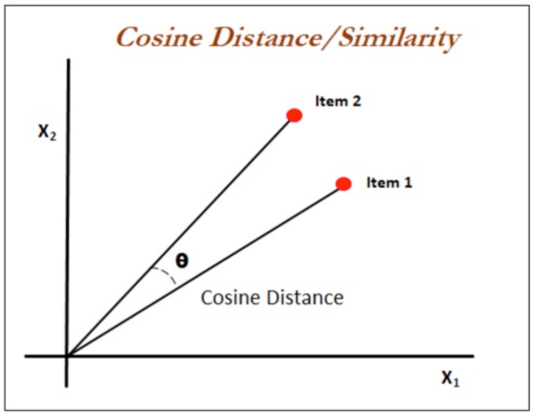

Data downloaded from Kaggle up to year 2017
Movie data web scraped from Wikipedia for year 2018-2022
Text preprocessing on movie data is done and converted into vectors
Distance between vectors is computed to find the closest vectors in order to find similar movies

Snapshot of data:



	genres	id	keywords	title	overview	cast	crew
0	[Action, Adventure, Fantasy, Science Fiction]	19995	[culture clash, future, space war, space colon...	Avatar	In the 22nd century, a paraplegic Marine is di...	[Sam Worthington, Zoe Saldana, Sigourney Weaver]	[{"credit_id": "52fe48009251416c750aca23", "de...
1	[Adventure, Fantasy, Action]	285	[ocean, drug abuse, exotic island, east india ...	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Johnny Depp, Orlando Bloom, Keira Knightley]	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	[Action, Adventure, Crime]	206647	[spy, based on novel, secret agent, sequel, mi...	Spectre	A cryptic message from Bond's past sends him o...	[Daniel Craig, Christoph Waltz, Léa Seydoux]	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	[Action, Crime, Drama, Thriller]	49026	[dc comics, crime fighter, terrorist, secret i...	The Dark Knight Rises	Following the death of District Attorney Harve...	[Christian Bale, Michael Caine, Gary Oldman]	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	[Action, Adventure, Science Fiction]	49529	[based on novel, mars, medallion, space travel...	John Carter	John Carter is a war-weary, former military ca...	[Taylor Kitsch, Lynn Collins, Samantha Morton]	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...
...	...	...	...	...	...	...	...
4800	[Action, Crime, Thriller]	9367	[united states–mexico barrier, legs, arms, pap...	El Mariachi	El Mariachi just wants to play his guitar and ...	[Carlos Gallardo, Jaime de Hoyos, Peter Marqua...	[{"credit_id": "52fe44eec3a36847f80b280b", "de...
4801	[Comedy, Romance]	72766	[]	Newlyweds	A newlywed couple's honeymoon is upended by th...	[Edward Burns, Kerry Bishé, Marsha Dietlein]	[{"credit_id": "52fe487dc3a368484e0fb013", "de...

How to find closest vectors : Here i chose cosine similarity method over Euclidean distance . Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance (due to the size of the document), chances are they may still be oriented closer together. The smaller the angle, higher the cosine similarity.

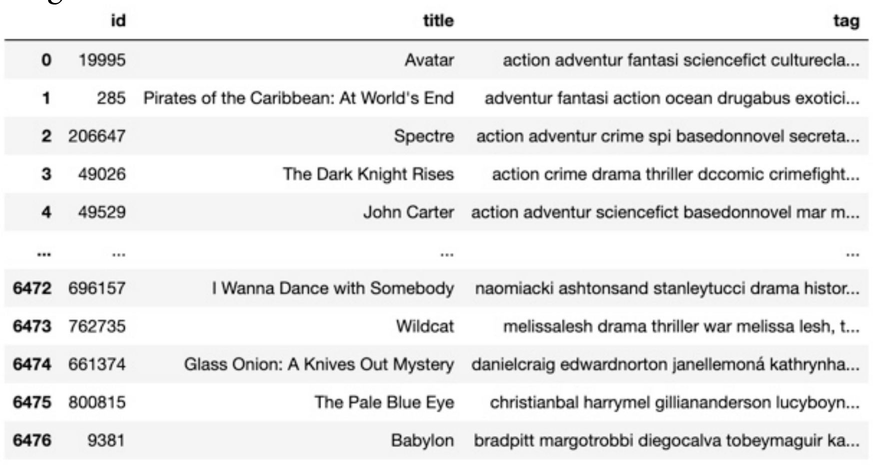


3. **Text Preprocessing:**

a. **Text Cleanup** : Basic preprocessing to remove special characters and html tags is done using regular expression

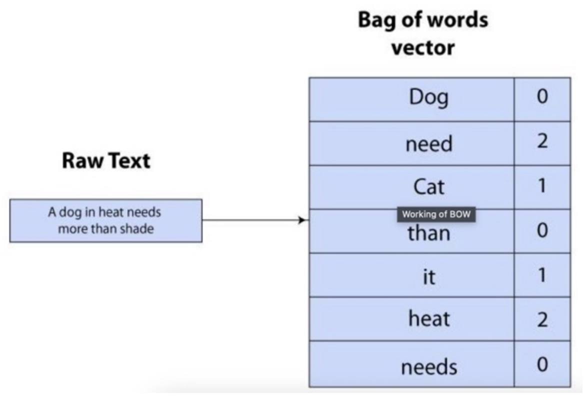
4. **Feature Engineering**:
    a. Director is extracted from crew column where job description is 'director'
    b. For cast we need only top three values which are most important for recommendation

c. All the values of cast, crew ,director and genres are joined into a new column called ‘tag’
d. Stemming is then done on ‘tag’ column to get the root word using PorterStemmer of nltk library of NLP.
e. Now each row contains ‘tag’ column which represents ‘cast’, ‘crew’, director’ and genres of each movie



	id	title	tag
0	19995	Avatar	action adventur fantasi sciencefict culturecla...
1	285	Pirates of the Caribbean: At World's End	adventur fantasi action ocean drugabus exotici...
2	206647	Spectre	action adventur crime spi basedonnovel secreta...
3	49026	The Dark Knight Rises	action crime drama thriller dccomic crimefight...
4	49529	John Carter	action adventur sciencefict basedonnovel mar m...
...	...	...	...
6472	696157	I Wanna Dance with Somebody	naomiacki ashtonsand stanleytucci drama histor...
6473	762735	Wildcat	melissalesh drama thriller war melissa lesh, t...
6474	661374	Glass Onion: A Knives Out Mystery	danielcraig edwardnorton janellemoná kathrynha...
6475	800815	The Pale Blue Eye	christianbal harrymel gilliananderson lucyboyn...
6476	9381	Babylon	bradpitt margotrobbi diegocalva tobeymaguir ka...

5. **Model building**:
Idea behind converting text into vectors:
All the words into ‘tag’ column of each movie are converted into a matrix called bag of words. For ex below is the BOW of text “A dog in heat needs more than shade”



a. This can be done using class sklearn.feature_extraction.text.CountVectorizer which would be in the shape of (rows=no. of rows in movie dataset , columns= total no. of unique words In ‘tag’ text which can be restricted to max_features)

b. Second method to convert the text 'tag' into vectors is Tfidf Vectorizer (Term Frequency Inverse Document Frequency) which calculates the term/word frequency of each unique word in the text.

Formula for calculating TF(term frequency):

Term frequency is defined as no. of times a word appear in a document (i) divided by total no. of words in the document(j):

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

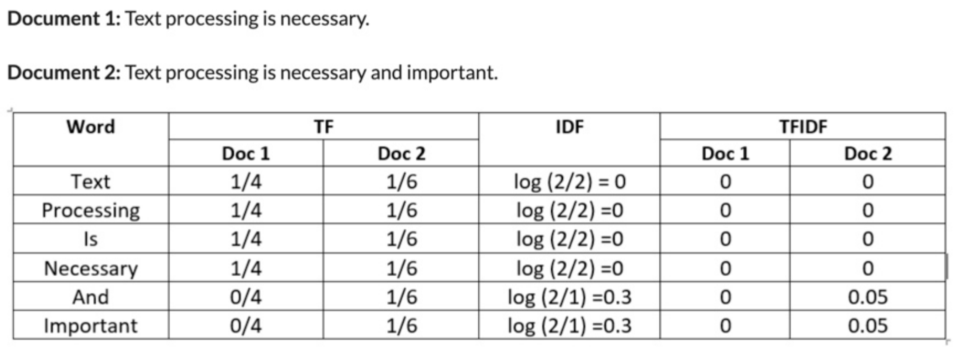
Formula for calculating Inverse document frequency :

Log of total no. of documents divided by no. of documents that contains the word:

$$idf(w) = log(\frac{N}{df_t})$$

Now the formula for TFidf is multiplying TF value with Idf value:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$



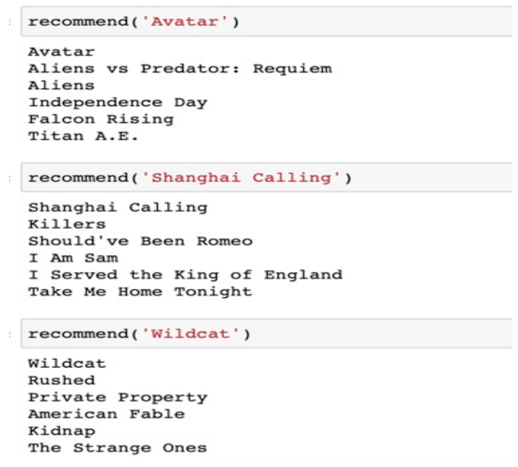
Document 1: Text processing is necessary.

Document 2: Text processing is necessary and important.

Word	TF		IDF	TFIDF	
	Doc 1	Doc 2		Doc 1	Doc 2
Text	1/4	1/6	log (2/2) = 0	0	0
Processing	1/4	1/6	log (2/2) =0	0	0
Is	1/4	1/6	log (2/2) =0	0	0
Necessary	1/4	1/6	log (2/2) =0	0	0
And	0/4	1/6	log (2/1) =0.3	0	0.05
Important	0/4	1/6	log (2/1) =0.3	0	0.05

c. Now the distance between the vectors is calculated using cosine_similarity using cosine_similarity method of sklearn.metrics.pairwise class .
In order to get the similar movies similarity score between vectors is sorted in descending order to get top few recommendations.

Results content based filtering:



```
recommend('Avatar')

Avatar
Aliens vs Predator: Requiem
Aliens
Independence Day
Falcon Rising
Titan A.E.

recommend('Shanghai Calling')

Shanghai Calling
Killers
Should've Been Romeo
I Am Sam
I Served the King of England
Take Me Home Tonight

recommend('Wildcat')

Wildcat
Rushed
Private Property
American Fable
Kidnap
The Strange Ones
```

2. Collaborative based movie recommender system

In this approach system filters out suggestions on the basis of similar users likes and reactions.System first looks at user likes and then adds up to find set of all other users with same taste. On the basis of that system recommends products ,movies they may like to watch, items they may like to buy etc.
For this approach I would need dataset that contains list of items and ratings given by other usres.

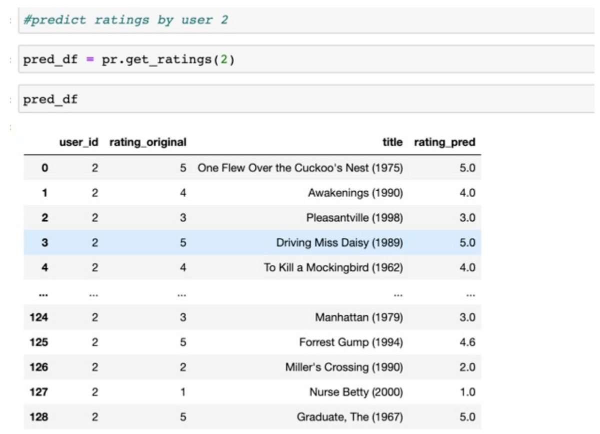
There are many ways to find similar users , two of them are listed below:

1. **First approach's :** here function gets the movie title and rating given to that movie by user as argument and recommend similar movies on the basis of rating given to that movie.
   a. If rating is high the function returns the movies which are highly correlated using Pearson correlation or cosine similarity score

b. If rating is low function returns those movies which are negatively correlated

2. **Second approach's has two functions**
    a. **First Function** : this approach predicts rating a user can give to movies.
    it gets user_id as argument and finds similar users using features like age, gender and genres and compute predicted rating as average of these user's ratings.

    b. **Second function** : this approach works almost same way but predicts ratings as weighted average of other's ratings on the basis of correlation coefficient .
    Steps followed:
    1. **Data Acquisition:** Movies, Ratings and Users Data is downloaded from https://grouplens.org/datasets/movielens/
    2. Here we are working on ratings of users so basically text preprocessing is not done in this approach
    3. Here we merge all the datasets : movies, ratings and users together to get data of all the user's and rating given to movies by those users.
    4. Now we get the similarity matrix of the above data using pivot table so that index contains the users_ids and column contains the features like age , gender and genres .
    5. With this similarity matrix we will get the similar users by computing the distance between each users by first converting into vectors whose dimensions are the features as above using cosine_similarity method of sklearn library.
    6. Based on this matrix movie rating is predicted by computing weighted average the ratings given by other users
    7. Snapshot of results for predicting movie ratings by user : 2



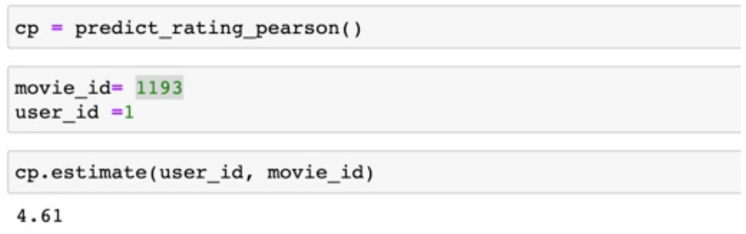
```
#predict ratings by user 2
```

```
pred_df = pr.get_ratings(2)
```

```
pred_df
```

	user_id	rating_original	title	rating_pred
0	2	5	One Flew Over the Cuckoo's Nest (1975)	5.0
1	2	4	Awakenings (1990)	4.0
2	2	3	Pleasantville (1998)	3.0
3	2	5	Driving Miss Daisy (1989)	5.0
4	2	4	To Kill a Mockingbird (1962)	4.0
...	...	...	...	...
124	2	3	Manhattan (1979)	3.0
125	2	5	Forrest Gump (1994)	4.6
126	2	2	Miller's Crossing (1990)	2.0
127	2	1	Nurse Betty (2000)	1.0
128	2	5	Graduate, The (1967)	5.0

8. Snapshot of results for predicting rating by user : 1 for movie_id : 1193:



```
cp = predict_rating_pearson()
```

```
movie_id= 1193
user_id =1
```

```
cp.estimate(user_id, movie_id)
```

```
4.61
```

Future improvements:

1. For more accuracy and better results surprise library which is Python scikit for recommender system can be used .
2.Results of both the models can be combined to get hybrid model recommender system