Movie Recommendation System

Abstract : There has been, in recent years, widespread interest in the topic of recommender systems that aid in the process of making selections from a wide space of alternatives. Recommender systems are agent-based systems that use stored user preferences to locate and suggest items that will be of interest to associated users. These systems will be useful and effective to the extent that they can make meaningful and consistent tradeoffs between con£icting user preferences. Research in this area has attracted the attention of people in the AI community,agent community and database community.

Develper's Contact Information :
Shubham Manikant Jha
shubhamjha.cse@gmail.com

Recommendation System :

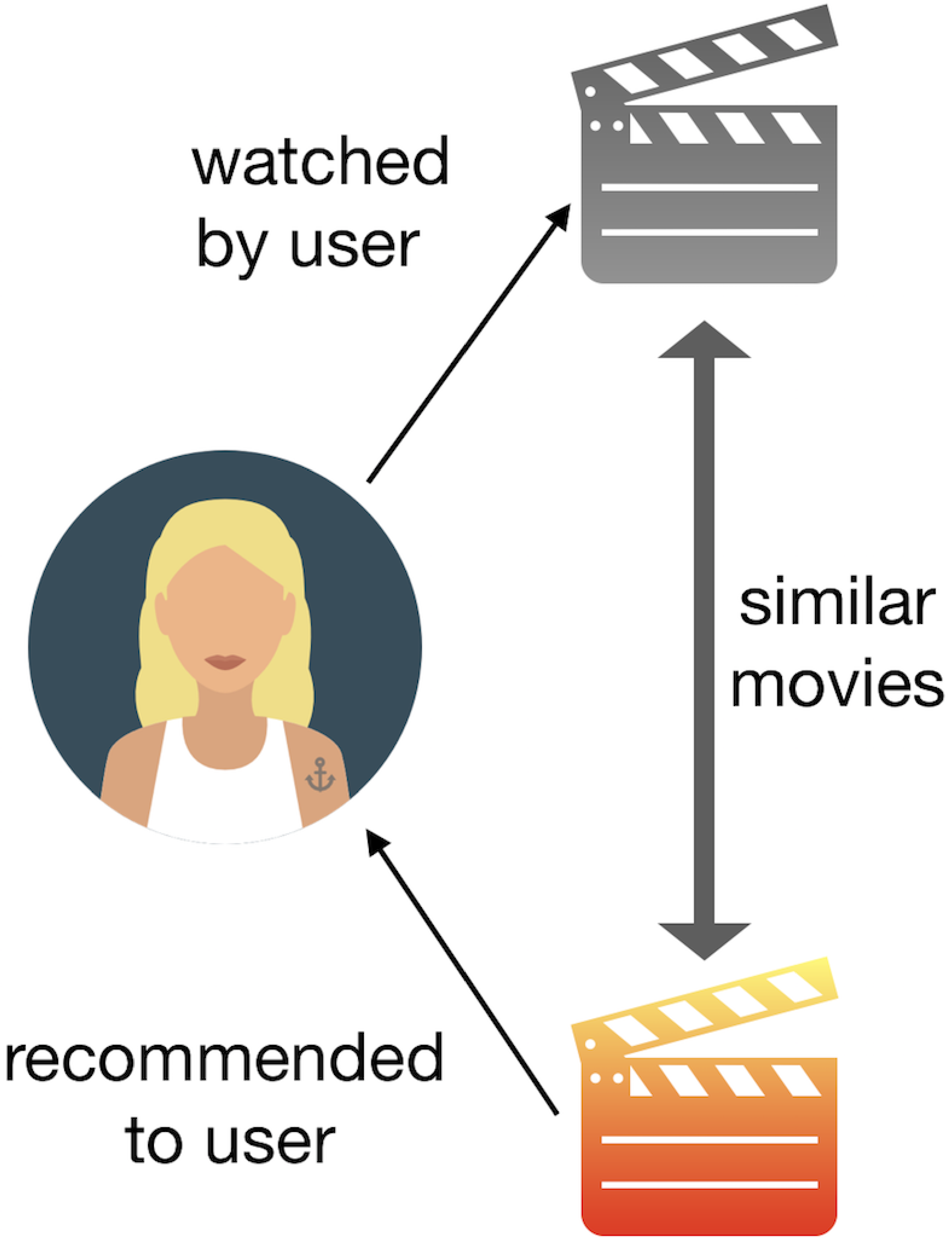
A recommendation system provides suggestions to the users through a filtering process that is based on user preferences and browsing history. The information about the user is taken as an input. The information is taken from the input that is in the form of browsing data. This information reflects the prior usage of the product as well as the assigned ratings. A recommendation system is a platform that provides its users with various contents based on their preferences and likings. A recommendation system takes the information about the user as an input. The recommendation system is an implementation of the machine learning algorithms

Types of Movie Recommendation systems

There are popularly five types of movie recommendation engine. They are :

1. Randomly Generated Recommendation engine
2. Popularity Based Recommendation engine without time bounds (Views count systems)
3. Popularity Based Recommendation engine with time bounds (Trending systems)
4. Content based recommendation engine
5. Collaborative filtering based recommendation engine

> *In this project, we will use Content based recommendation engine. We will implement this system by using Machine Learning Algorithm in Python. For that purpose, we will import Scikit-learn library.*



About the Dataset : The movie dataset is made from IMDB. It is in the form of CSV file.

Content based recommendation engine

Importing the packages

- pandas : for reading and analysing the csv file.
- scikit-learn : for implementing the algorithms

In [2]:
```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
```

Reading the CSV File

In [15]:
```
df = pd.read_csv("data/IMDB_dataset.csv")
print("Sample :")
df.head()
```

```
Sample :
```

Out[15]:

	index	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	19995	culture clash future space war space colony so...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285	ocean drug abuse exotic island east india trad...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647	spy based on novel secret agent sequel mi6	en	Spectre	A cryptic message from Bond’s past sends him o...	107.376788
3	3	250000000	Action Crime Drama Thriller	http://www.thedarkknightrises.com/	49026	dc comics crime fighter terrorist secret ident...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950
4	4	260000000	Action Adventure Science Fiction	http://movies.disney.com/john-carter	49529	based on novel mars medallion space travel pri...	en	John Carter	John Carter is a war-weary, former military ca...	43.926995

5 rows × 24 columns

In [9]:
```
print("Listing out Features :")
df.columns
```

```
Listing out Features :
```

Out[9]:
```
Index(['index', 'budget', 'genres', 'homepage', 'id', 'keywords',
       'original_language', 'original_title', 'overview', 'popularity',
       'production_companies', 'production_countries', 'release_date',
       'revenue', 'runtime', 'spoken_languages', 'status', 'tagline', 'title',
       'vote_average', 'vote_count', 'cast', 'crew', 'director'],
      dtype='object')
```

Feature Engineering : We will extract "keywords", as it plays an important role in similarity assosiation. Added to that, a person who watches a movie is more likely to watch movies having same "cast" and "director". People tend to watch the movies which fit in their "genres".

Conclusion : We will extract the following features :

- Keywords
- Cast
- Generes
- Director

In [10]:
```
features = ['keywords','cast','genres','director']
```

Combining features

In [17]:
```
def collect_features(row):
    return row['keywords']+" "+row['cast']+" "+row['genres']+" "+row['director']

for feature in features:
    df[feature] = df[feature].fillna('')

df["combined_features"] = df.apply(collect_features,axis=1)
```

Sample of combined feature :
0 culture clash future space war space colony so...
1 ocean drug abuse exotic island east india trad...
2 spy based on novel secret agent sequel mi6 Dan...
3 dc comics crime fighter terrorist secret ident...
4 based on novel mars medallion space travel pri...

Vectorizing the text

Scikit-learn's CountVectorizer is used to transform a corpora of text to a vector of term /token counts. It also provides the capability to preprocess the text data prior to generating the vector representation making it a highly flexible feature representation module for text.

In [19]:
```
cv = CountVectorizer()
count_matrix = cv.fit_transform(df["combined_features"])
```

Making a Cosine-Similarity Matrix

A direction cosine matrix (DCM) is a transformation matrix that transforms one coordinate reference frame to another.

Cosine similarity is a metric used to determine how similar two entities are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space.

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\|\vec{b}\|}$$

$$\|\vec{a}\| = \sqrt{a_1^2 + a_2^2 + a_3^2 + \cdots + a_n^2}$$

$$\|\vec{b}\| = \sqrt{b_1^2 + b_2^2 + b_3^2 + \cdots + b_n^2}$$

In [20]:
```
similarity_matrix = cosine_similarity(count_matrix)
```

Here we will define two function which takes index and return movie title and vice-versa

In [21]:
```
def get_title_from_index(index):
    return df[df.index == index]["title"].values[0]
def get_index_from_title(title):
    return df[df.title == title]["index"].values[0]
```

Not we will feed the system, the videos user has watched

In [41]:
```
watched_video = "Avatar"
```

In [42]:

```
movie_index = get_index_from_title(watched_video)
similar_movies = list(enumerate(similarity_matrix[movie_index]))
```

We will sort the list similar_movies according to similarity scores in descending order. Since the most similar movie to a given movie will be itself, we will discard the first element after sorting the movies.

In [43]:

```
sorted_similar_movies = sorted(similar_movies,key=lambda x:x[1],reverse=True)[1:]
```

Movies you may like

Here we will print the first few movies that are similar to the movies the user has watches

In [46]:

```
count=0
print("Movies you may like :")
for movie in sorted_similar_movies:
    print(get_title_from_index(movie[0]))
    count=count+1
    if count>7:
        break
```

```
Movies you may like :
Guardians of the Galaxy
Aliens
Star Wars: Clone Wars: Volume 1
Star Trek Into Darkness
Star Trek Beyond
Alien
Lockout
Jason X
```

Results :

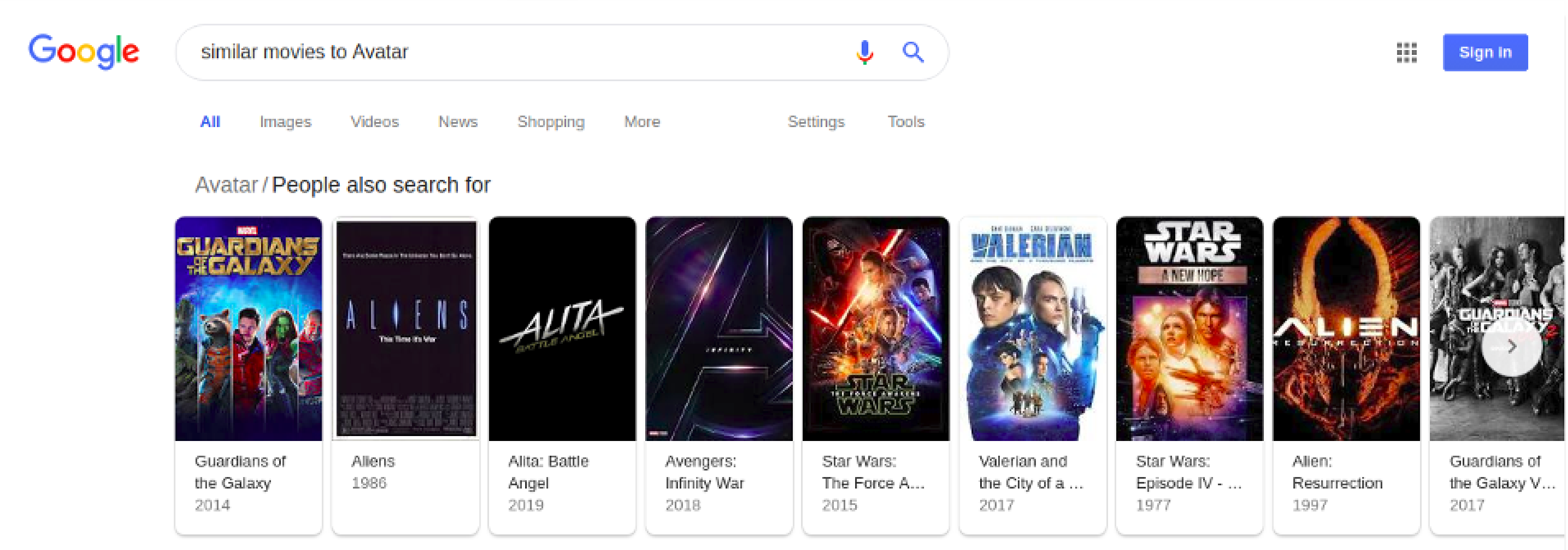
Movie watched by user :

- Avatar

Movie user might like :

- Guardians of the Galaxy
- Aliens
- Star Wars: Clone Wars: Volume 1
- Star Trek Into Darkness
- Star Trek Beyond
- Alien
- Lockout
- Jason X

Let us see what google has to say about users who watch this movie :



similar movies to Avatar

Avatar / People also search for

Guardians of the Galaxy 2014

Aliens 1986

Alita: Battle Angel 2019

Avengers: Infinity War 2018

Star Wars: The Force A... 2015

Valerian and the City of a ... 2017

Star Wars: Episode IV - ... 1977

Alien: Resurrection 1997

Guardians of the Galaxy V... 2017