Types of Recommendar System:

1. Content RS
2. Collaborative RS
3. Hybrid RS

1. Content RS: Basis on Content Similarities, the Recommendar System suggests or provides content.
2. Collaborative RS: Basis on Similarities of some other content consumer, the RS provides the same content to user. (Matching Contents)
3. Hybrid RS: It is basically combination of Content RS and Collaborative RS.

This Movie Recommender System is Content Based RS.

```
# Importing Libraries
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
import nltk
from sklearn.metrics.pairwise import cosine_similarity
import pickle
```

```
# Downloading necessary Files and Datasets
! cp kaggle.json ~/.kaggle/
```

```
! chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets download -d tmdb/tmdb-movie-metadata
```

```
Downloading tmdb-movie-metadata.zip to /content
 56% 5.00M/8.89M [00:00<00:00, 32.2MB/s]
100% 8.89M/8.89M [00:00<00:00, 51.7MB/s]
```

```
!unzip /content/tmdb-movie-metadata.zip
```

```
Archive:  /content/tmdb-movie-metadata.zip
  inflating: tmdb_5000_credits.csv
  inflating: tmdb_5000_movies.csv
```

As we can see in Dataset we have 2 different csv Files.

First File is Movies file in which Data such as Budget, Genres, Homepage link, Movie_id, Keywords, Original_title, Overview, Popularity, Companies, Countries, Release_date, Revenue, Runtime, Spoken_Language, Status, Taglines, Votes are mentioned. There are many features which will not be utilised so dropping them in further steps.

Second File consists are columns such Movies_id, Movies_Titles, Casts and Crew. This dataframe has less columns so Feature Engineering is Required on this Dataframe to utilise best data out of it.

```
movies_df = pd.read_csv("/content/tmdb_5000_movies.csv")
```

```
credits_df = pd.read_csv("/content/tmdb_5000_credits.csv")
```

```
movies_df.head(1)
```

```
ds  original_language  original_title  overview  popularity  production_companies  production_countries  release_date  revenue  runtim
```

```
credits_df.head(1)
```

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...

Merging both dataframes on the basis of Title.

```
movies = movies_df.merge(credits_df,on='title')
```

```
movies.head(2)
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615

2 rows × 23 columns

```
movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4809 entries, 0 to 4808
Data columns (total 23 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   budget                4809 non-null   int64  
 1   genres                4809 non-null   object 
 2   homepage              1713 non-null   object 
 3   id                    4809 non-null   int64  
 4   keywords              4809 non-null   object 
 5   original_language     4809 non-null   object 
 6   original_title        4809 non-null   object 
 7   overview              4806 non-null   object 
 8   popularity            4809 non-null   float64
 9   production_companies  4809 non-null   object 
 10  production_countries  4809 non-null   object 
 11  release_date          4808 non-null   object 
 12  revenue               4809 non-null   int64  
 13  runtime               4807 non-null   float64
 14  spoken_languages      4809 non-null   object 
 15  status                4809 non-null   object 
 16  tagline               3965 non-null   object 
 17  title                 4809 non-null   object 
 18  vote_average          4809 non-null   float64
 19  vote_count            4809 non-null   int64  
 20  movie_id              4809 non-null   int64  
 21  cast                  4809 non-null   object 
 22  crew                  4809 non-null   object 
dtypes: float64(3), int64(5), object(15)
memory usage: 901.7+ KB
```

As from Above data we can see that there are no null data found in Dataframe.

```
movies.columns
```

```
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', 'movie_id', 'cast', 'crew'],
      dtype='object')
```

Dropping Unwanted Columns:

1. Budget
2. Homepage
3. Original_Language
4. Original Title
5. Popularity
6. Production Companies/ Production

```
movies.shape
```

```
(4809, 23)
```

```
movies = movies[['movie_id','title','genres','keywords','overview','cast', 'crew']]
```

```
movies.head(2)
```

	movie_id	title	genres	keywords	overview	cast	crew
0	19995	Avatar	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 1463, "name": "culture clash"}, {"id":...	In the 22nd century, a paraplegic Marine is di...	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At	[{"id": 12, "name": "Adventure"}, {"id":	[{"id": 270, "name": "ocean"}, {"id":	Captain Barbossa, long believed to be	[{"cast_id": 4, "character": "Captain	[{"credit_id": "52fe4232c3a36847f800b579",

```
movies.isnull().sum()
```

```
movie_id    0
title       0
genres      0
keywords    0
overview    3
cast        0
crew        0
dtype: int64
```

```
movies.duplicated().sum()
```

```
0
```

Feature Engineering

```
import ast
def conversion(cols):
  converted = []
  for i in ast.literal_eval(cols):
    converted.append(i['name'])
  return converted
```

```
movies['genres'] = movies['genres'].apply(conversion)
```

```
movies['keywords'] = movies['keywords'].apply(conversion)
```

```
def conversion_top_3(cols):
  converted = []
  counter = 0
  for i in ast.literal_eval(cols):
    if counter !=3:
      converted.append(i['name'])
      counter+=1
    else:
      break
  return converted
```

```
movies['cast'] = movies['cast'].apply(conversion_top_3)
```

```
movies.head()
```

	movie_id	title	genres	keywords	overview	cast	crew
0	19995	Avatar	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	In the 22nd century, a paraplegic Marine is di...	[Sam Worthington, Zoe Saldana, Sigourney Weaver]	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...	Captain Barbossa, long believed to be dead, ha...	[Johnny Depp, Orlando Bloom, Keira Knightley]	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, mi...	A cryptic message from Bond's past sends him o...	[Daniel Craig, Christoph Waltz, Léa Seydoux]	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret i...	Following the death of District Attorney Harve...	[Christian Bale, Michael Caine, Gary Oldman]	[{"credit_id": "52fe4781c3a36847f81398c3", "de...

```
movies.crew[0]
```

```
'[{"credit_id": "52fe48009251416c750aca23", "department": "Editing", "gender": 0, "id": 1721, "job": "Editor", "name": "Stephen E. Rivk
in"}, {"credit_id": "539c47ecc3a36810e3001f87", "department": "Art", "gender": 2, "id": 496, "job": "Production Design", "name": "Rick
Carter"}, {"credit_id": "54491c89c3a3680fb4001cf7", "department": "Sound", "gender": 0, "id": 900, "job": "Sound Designer", "name": "Ch
ristopher Boyes"}, {"credit_id": "54491cb70e0a267480001bd0", "department": "Sound", "gender": 0, "id": 900, "job": "Supervising Sound E
ditor", "name": "Christopher Boyes"}, {"credit_id": "539c4a4cc3a36810c9002101", "department": "Production", "gender": 1, "id": 1262, "j
ob": "Casting", "name": "Mali Finn"}, {"credit_id": "5544ee3b925141499f0008fc", "department": "Sound", "gender": 2, "id": 1729, "job":
```

```
def fetch_dir(cols):
  dir_name = []
  for i in ast.literal_eval(cols):
    if i['job'] == 'Director':
      dir_name.append(i['name'])
  return dir_name
```

```
movies['crew'] = movies.crew.apply(fetch_dir)
```

```
movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4809 entries, 0 to 4808
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   movie_id  4809 non-null   int64 
 1   title     4809 non-null   object
 2   genres    4809 non-null   object
 3   keywords  4809 non-null   object
 4   overview  4806 non-null   object
 5   cast      4809 non-null   object
 6   crew      4809 non-null   object
dtypes: int64(1), object(6)
memory usage: 429.6+ KB
```

```
movies['overview'] = movies['overview'].astype('str')
```

```
movies['overview'] = movies['overview'].apply(lambda x:x.split())
```

```
movies.head()
```

	movie_id	title	genres	keywords	overview	cast	crew
			[Action, Adventure,	[culture clash, future,	[In, the, 22nd, century,	[Sam Worthington, Zoe	[James
		Pirates of the ... End		[ocean, drug abuse ...	[Captain, Barbossa ... d...		
2	206647	Spectre	Crime]	secret agent, sequel, mi	from, Bond's, past, send	Waltz, Léa Seydoux]	[Sam Mendes]
3	49026	The Dark Knight	[Action, Crime, Drama,	fighter, terrorist, secret	[Following, the, death,	[Christian Bale, Michael	[Christopher

```
movies['genres'] = movies['genres'].apply(lambda x:[i.replace(" ","") for i in x])
```

```
movies['keywords'] = movies['keywords'].apply(lambda x:[i.replace(" ","") for i in x])
```

```
movies['cast'] = movies['cast'].apply(lambda x:[i.replace(" ","") for i in x])
```

```
movies['crew'] = movies['crew'].apply(lambda x:[i.replace(" ","") for i in x])
```

```
movies['tag'] = movies['overview']+movies['cast']+movies['crew']+movies['genres']+movies['keywords']
```

```
new_df = movies[['movie_id','title','tag']]
```

```
new_df['tag'][0]
```

```
['In',
 'the',
 '22nd',
 'century,',
 'a',
 'paraplegic',
 'Marine',
 'is',
 'dispatched',
 'to',
 'the',
 'moon',
 'Pandora',
 'on',
 'a',
 'unique',
 'mission,',
 'but',
 'becomes',
 'torn',
 'between',
 'following',
 'orders',
 'and',
 'protecting',
 'an',
 'alien',
 'civilization.',
 'SamWorthington',
 'ZoeSaldana',
 'SigourneyWeaver',
 'JamesCameron',
 'Action',
 'Adventure',
 'Fantasy',
 'ScienceFiction',
 'cultureclash',
 'future',
 'spacewar',
 'spacecolony',
 'society',
 'spacetravel',
 'futuristic',
 'romance',
 'space',
 'alien',
 'tribe',
 'alienplanet',
 'cgi',
 'marine',
 'soldier',
 'battle',
 'loveaffair',
 'antiwar',
 'powerrelations',
 'mindandsoul',
 '3d']
```

```
new_df['tag'] = new_df['tag'].apply(lambda x:" ".join(x))
```

```
<ipython-input-45-9e0646ac0df3>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  new_df['tag'] = new_df['tag'].apply(lambda x:" ".join(x))
```

```
new_df.head(2)
```

	movie_id	title	tag
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...

```
new_df['tag'][0]
```

```
'In the 22nd century, a paraplegic Marine is dispatched to the moon Pandora on a unique mission, but becomes torn between following ord
ers and protecting an alien civilization. SamWorthington ZoeSaldana SigourneyWeaver JamesCameron Action Adventure Fantasy ScienceFictio
n cultureclash future spacewar spacecolony society spacetravel futuristic romance space alien tribe alienplanet cgi marine soldier batt
le loveaffair antiwar powerrelations mindandsoul 3d'
```

```
new_df['tag'] = new_df['tag'].apply(lambda x:x.lower())
```

```
<ipython-input-48-f9c79f0acdd8>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  new_df['tag'] = new_df['tag'].apply(lambda x:x.lower())
```

```
new_df.head()
```

	movie_id	title	tag
0	19995	Avatar	in the 22nd century, a paraplegic marine is di...
1	285	Pirates of the Caribbean: At World's End	captain barbossa, long believed to be dead, ha...
2	206647	Spectre	a cryptic message from bond's past sends him o...
3	49026	The Dark Knight Rises	following the death of district attorney harve...
4	49529	John Carter	john carter is a war-weary, former military ca...

Modelling

```
cv = CountVectorizer(max_features =5000 ,stop_words = 'english')
```

```
vectors = cv.fit_transform(new_df['tag']).toarray()
```

```
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
```

```
def stem(txt):
  aft_stemming = []
  for i in txt.split():
    aft_stemming.append(ps.stem(i))
  return " ".join(aft_stemming)
```

```
new_df['tag'] = new_df['tag'].apply(stem)
```

```
<ipython-input-58-c98a7ce51958>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
    See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
      new_df['tag'] = new_df['tag'].apply(stem)
```

```
cv = CountVectorizer(max_features =5000 ,stop_words = 'english')

vectors = cv.fit_transform(new_df['tag']).toarray()
```

```
similarity = cosine_similarity(vectors)
```

```
similarity[0]
```

```
    array([1.        , 0.08346223, 0.0860309 , ..., 0.04543109, 0.        ,
           0.        ])
```

```
def recommend(movie):
    index = new_df[new_df['title'] == movie].index[0]
    distances = sorted(list(enumerate(similarity[index])),reverse=True,key = lambda x: x[1])
    for i in distances[1:6]:
        print(new_df.iloc[i[0]].title)
```

Recommender System

```
recommend('Batman')
```

```
    Batman
    Batman & Robin
    Batman Begins
    Batman Returns
    The R.M.
```

```
recommend('Avatar')
```

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

Final Wordings:

1. The Dataset consists of around 5000 datapoints.
2. The Recommendation System Used in this is Content Based Recommendation System, which detects and finds similar users and suggest similar contents.
3. In process we have dropped few features for better runnabiltiy and for better usability of Data, so with that features we can also try some other recommender system
4. In Feature Engineering, We can converted, merged few columns which have similar functionality and which results in a combined Dataframe.
5. Some NLP concepts are used in this projects on the basis of which we are calculating similar contents.
6. Cosine Similarity is used as distance calculator between the content as Euclidian Distance would have cause Curse of Dimensionality issues, since that's why Cosine Similarity is used. Instead of Cosine Similarity, Cosine Distance can also be used.
7. From above point, we have made a Recommender System which we can see runs smoothly and also giving suggestions based on Keywords.