

#MOVIE RECOMENDATION SYSTEM

#import libraries

import pandas as pd
import numpy as np

#import dataset

df = pd.read_csv('/content/Movies Recommendation.csv')

df.head()

	Movie_ID	Movie_Title	Movie_Genre	Movie_Language	Movie_Budget	Movie_Popularity	M
0	1	Four Rooms	Crime Comedy	en	4000000	22.876230	
1	2	Star Wars	Adventure Action Science Fiction	en	11000000	126.393695	
2	3	Finding Nemo	Animation Family	en	94000000	85.688789	
3	4	Forrest Gump	Comedy Drama Romance	en	55000000	138.133331	
4	5	American Beauty	Drama	en	15000000	80.878605	

5 rows × 21 columns

df.info()

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 4760 entries, 0 to 4759  
Data columns (total 21 columns):  
#   Column                                Non-Null Count  Dtype  
---  -  
0   Movie_ID                             4760 non-null   int64  
1   Movie_Title                           4760 non-null   object  
2   Movie_Genre                           4760 non-null   object  
3   Movie_Language                         4760 non-null   object  
4   Movie_Budget                           4760 non-null   int64  
5   Movie_Popularity                       4760 non-null   float64  
6   Movie_Release_Date                     4760 non-null   object  
7   Movie_Revenue                           4760 non-null   int64  
8   Movie_Runtime                           4758 non-null   float64  
9   Movie_Vote                             4760 non-null   float64  
10  Movie_Vote_Count                       4760 non-null   int64  
11  Movie_Homepage                         1699 non-null   object  
12  Movie_Keywords                         4373 non-null   object  
13  Movie_Overview                         4757 non-null   object  
14  Movie_Production_House                 4760 non-null   object  
15  Movie_Production_Country               4760 non-null   object  
16  Movie_Spoken_Language                  4760 non-null   object  
17  Movie_Tagline                          3942 non-null   object
```

```
18 Movie_Cast          4733 non-null object
19 Movie_Crew          4760 non-null object
20 Movie_Director      4738 non-null object
dtypes: float64(3), int64(4), object(14)
memory usage: 781.1+ KB

df.shape

(4760, 21)

df.columns

Index(['Movie_ID', 'Movie_Title', 'Movie_Genre', 'Movie_Language',
      'Movie_Budget', 'Movie_Popularity', 'Movie_Release_Date',
      'Movie_Revenue', 'Movie_Runtime', 'Movie_Vote', 'Movie_Vote_Count',
      'Movie_Homepage', 'Movie_Keywords', 'Movie_Overview',
      'Movie_Production_House', 'Movie_Production_Country',
      'Movie_Spoken_Language', 'Movie_Tagline', 'Movie_Cast', 'Movie_Crew',
      'Movie_Director'],
      dtype='object')

#Get feature Selection

df_features= df [['Movie_Genre','Movie_Keywords','Movie_Tagline','Movie_Cast','Movie_Director']].fillna('')

df_features.shape

(4760, 5)

df_features



|      | Movie_Genre                      | Movie_Keywords                                    | Movie_Tagline                                     | Movie_Cast                                        | Movie_Director  |
|------|----------------------------------|---------------------------------------------------|---------------------------------------------------|---------------------------------------------------|-----------------|
| 0    | Crime Comedy                     | hotel new year's eve witch bet hotel room         | Twelve outrageous guests. Four scandalous requ... | Tim Roth Antonio Banderas Jennifer Beals Madon... | Allison Anders  |
| 1    | Adventure Action Science Fiction | android galaxy hermit death star lightsaber       | A long time ago in a galaxy far, far away...      | Mark Hamill Harrison Ford Carrie Fisher Peter ... | George Lucas    |
| 2    | Animation Family                 | father son relationship harbor underwater fish... | There are 3.7 trillion fish in the ocean, they... | Albert Brooks Ellen DeGeneres Alexander Gould ... | Andrew Stanton  |
| 3    | Comedy Drama Romance             | vietnam veteran hippie mentally disabled runni... | The world will never be the same, once you've ... | Tom Hanks Robin Wright Gary Sinise Mykelti Wil... | Robert Zemeckis |
| 4    | Drama                            | male nudity female nudity adultery midlife cri... | Look closer.                                      | Kevin Spacey Annette Bening Thora Birch Wes Be... | Sam Mendes      |
| ...  | ...                              | ...                                               | ...                                               | ...                                               | ...             |
| 4755 | Horror                           |                                                   | The hot spot where Satan's waitin'.               | Lisa Hart Carroll Michael Des Barres Paul Drak... | Pece Dingo      |
| 4756 | Comedy Family Drama              |                                                   | It's better to stand out than to fit in.          | Roni Akurati Brighton Sharbino Jason Lee Anjul... | Frank Lotito    |
| 4757 | Thriller Drama                   | christian film sex trafficking                    | She never knew it could happen to her...          | Nicole Smolen Kim Baldwin Ariana Stephens Brys... | Jaco Booyens    |
| 4758 | Family                           |                                                   |                                                   |                                                   |                 |



Next steps: Generate code with df\_features View recommended plots

X = df_features['Movie_Genre'] +' '+df_features['Movie_Keywords']+' '+df_features['Movie_Tagline']+' '+df_features['Movie_Cast']+' '+df_feat

X

0 Crime Comedy hotel new year's eve witch bet ho...
1 Adventure Action Science Fiction android galax...
2 Animation Family father son relationship harbo...
3 Comedy Drama Romance vietnam veteran hippie me...
4 Drama male nudity female nudity adultery midli...
...
4755 Horror The hot spot where Satan's waitin'. Li...
4756 Comedy Family Drama It's better to stand out ...
4757 Thriller Drama christian film sex trafficking ...
```

```

4758                                     Family
4759 Documentary music actors legendary performer cl...
Length: 4760, dtype: object

```

```
X.shape
```

```
(4760,)
```

```
#GET FEATURE TEXT CONVERSION TO TOKENS
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
tfidf = TfidfVectorizer()
```

```
X = tfidf.fit_transform(X)
```

```
X.shape
```

```
(4760, 17258)
```

```
print(X)
```

```

(0, 617)      0.1633382144407513
(0, 492)      0.1432591540388685
(0, 15413)    0.1465525095337543
(0, 9675)     0.14226057295252661
(0, 9465)     0.1659841367820977
(0, 1390)     0.16898383612799558
(0, 7825)     0.09799561597509843
(0, 1214)     0.13865857545144072
(0, 729)      0.13415063359531618
(0, 13093)    0.1432591540388685
(0, 15355)    0.10477815972666779
(0, 9048)     0.0866842116160778
(0, 11161)    0.06250380151644369
(0, 16773)    0.17654247479915475
(0, 5612)     0.08603537588547631
(0, 16735)    0.10690083751525419
(0, 7904)     0.13348000542112332
(0, 15219)    0.09800472886453934
(0, 11242)    0.07277788238484746
(0, 3878)     0.11998399582562203
(0, 5499)     0.11454057510303811
(0, 7071)     0.19822417598406614
(0, 7454)     0.14745635785412262
(0, 1495)     0.19712637387361423
(0, 9206)     0.15186283580984414
:
(4757, 5455)  0.12491480594769522
(4757, 2967)  0.16273475835631626
(4757, 8464)  0.23522565554066333
(4757, 6938)  0.17088173678136628
(4757, 8379)  0.17480603856721913
(4757, 15303) 0.07654356007668191
(4757, 15384) 0.09754322497537371
(4757, 7649)  0.11479421494340192
(4757, 10896) 0.14546473055066447
(4757, 4494)  0.05675298448720501
(4758, 5238)  1.0
(4759, 11264) 0.33947721804318337
(4759, 11708) 0.33947721804318337
(4759, 205)   0.3237911628497312
(4759, 8902)  0.3040290704566037
(4759, 14062) 0.3237911628497312
(4759, 3058)  0.2812896191863103
(4759, 7130)  0.26419662449963793
(4759, 10761) 0.3126617295732147
(4759, 4358)  0.18306542312175342
(4759, 14051) 0.20084315377640435
(4759, 5690)  0.19534291014627303
(4759, 15431) 0.19628653185946862
(4759, 1490)  0.21197258705292082
(4759, 10666) 0.15888268987343043

```

```
#Get similarity score using cosine similarity
```

```
from sklearn.metrics.pairwise import cosine_similarity
```

```
similarity_score = cosine_similarity(X)
```

```
similarity_score
```

```
array([[1.          , 0.01351235, 0.03570468, ..., 0.          , 0.          ,
        0.          ],
       [0.01351235, 1.          , 0.00806674, ..., 0.          , 0.          ,
        0.          ],
       [0.03570468, 0.00806674, 1.          , ..., 0.          , 0.08014876,
        0.          ],
       ...,
       [0.          , 0.          , 0.          , ..., 1.          , 0.          ,
        0.          ],
       [0.          , 0.          , 0.08014876, ..., 0.          , 1.          ,
        0.          ],
       [0.          , 0.          , 0.          , ..., 0.          , 0.          ,
        1.          ]])
```

```
similarity_score.shape
```

```
(4760, 4760)
```

```
#Get movie name as input from user and validate for closest spelling
```

```
Favourite_Movie_Name = input('Enter your favourite movie name:')
```

```
Enter your favourite movie name:avtaar
```

```
All_Movies_Title_List = df['Movie_Title'].tolist()
```

```
import difflib
```

```
Movie_Recommendation = difflib.get_close_matches(Favourite_Movie_Name,All_Movies_Title_List)
print(Movie_Recommendation)
```

```
['Avatar', 'Gattaca']
```

```
Close_Match = Movie_Recommendation[0]
print(Close_Match)
```

```
Avatar
```

```
Start coding or generate with AI.
```

```
Index_of_Close_Match_Movie = df[df.Movie_Title == Close_Match]['Movie_ID'].values[0]
print(Index_of_Close_Match_Movie)
```

```
2692
```

```
#Getting a list of similar movies
Recommendation_Score = list(enumerate(similarity_score[Index_of_Close_Match_Movie]))
print(Recommendation_Score)
```

```
0, 0.009805093506053453), (1, 0.0), (2, 0.0), (3, 0.00800429043895183), (4, 0.0026759665928032302), (5, 0.009639835665946627), (6, 0.0045
```

```
len(Recommendation_Score)
```

```
4760
```

```
# get all movies sort based on recommendation score wrt favourite movie
```

```
# sorting the movies based on their similarity score
Sorted_Similar_Movies = sorted(Recommendation_Score, key = lambda X:X[1] , reverse=True)
print(Sorted_Similar_Movies)
```

```
[(2692, 1.0000000000000002), (3276, 0.11904275527845871), (3779, 0.10185805797079382), (62, 0.10153560702418994), (2903, 0.1006378731438
```

```
# print the names of similar movies based on their index
print('Top 30 movies suggested for you :\n')
i=1
for movie in Sorted_Similar_Movies:
    index = movie[0]
    title_from_index = df[df.index == index]['Movie_Title'].values[0]
    if(i<31):
        print(i, '.', title_from_index)
        i+=1
```

Top 30 movies suggested for you :

```
1 . Niagara
2 . Caravans
3 . My Week with Marilyn
4 . Brokeback Mountain
5 . Harry Brown
6 . Night of the Living Dead
7 . The Curse of Downers Grove
8 . The Boy Next Door
9 . Back to the Future
10 . The Juror
11 . Some Like It Hot
12 . Enough
13 . The Kentucky Fried Movie
14 . Eye for an Eye
15 . Welcome to the Sticks
16 . Alice Through the Looking Glass
17 . Superman III
18 . The Misfits
19 . Premium Rush
20 . Duel in the Sun
21 . Sabotage
22 . Small Soldiers
23 . All That Jazz
24 . Camping Sauvage
25 . The Raid
26 . Beyond the Black Rainbow
27 . To Kill a Mockingbird
28 . World Trade Center
29 . The Dark Knight Rises
30 . Tora! Tora! Tora!
```

```
# top 10 movie recommendation system
Movie_Name = input('Enter your favourite movie name:')
list_of_all_titles = df['Movie_Title'].tolist()
Find_Close_Match = difflib.get_close_matches(Favourite_Movie_Name, All_Movies_Title_List)
Close_Match = Find_Close_Match[0]
Index_of_Movie = df[df.Movie_Title == Close_Match]['Movie_ID'].values[0]
Recommendation_Score = list(enumerate(similarity_score[Index_of_Close_Match_Movie]))
Sorted_Similar_Movies = sorted(Recommendation_Score, key = lambda X:X[1] , reverse=True)
print('Top 10 movies suggested for you :\n')
i=1
for movie in Sorted_Similar_Movies:
    index = movie[0]
    title_from_index = df[df.index == index]['Movie_Title'].values[0]
    if(i<11):
        print(i, '.', title_from_index)
        i+=1
```

Enter your favourite movie name:avtaar
Top 10 movies suggested for you :

```
1 . Niagara
2 . Caravans
3 . My Week with Marilyn
4 . Brokeback Mountain
5 . Harry Brown
6 . Night of the Living Dead
7 . The Curse of Downers Grove
8 . The Boy Next Door
9 . Back to the Future
10 . The Juror
```

