In [1]:
```
import pandas as pd
import numpy as np
import matplotlib
from matplotlib import pyplot as plt
%matplotlib inline
import difflib
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
```

In [2]:
```
movie_data = pd.read_csv('C:\\Users\\Atif\\Desktop\\python learning\\movies.csv')
movie_data.head()
```

Out[2]:

	index	budget	genres	homepage	id	keywords	original_language
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	19995	culture clash future space war space colony so...	en
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285	ocean drug abuse exotic island east india trad...	en
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647	spy based on novel secret agent sequel mi6	en
3	3	250000000	Action Crime Drama Thriller	http://www.thedarkknightrises.com/	49026	dc comics crime fighter terrorist secret ident...	en
4	4	260000000	Action Adventure Science Fiction	http://movies.disney.com/john-carter	49529	based on novel mars medallion space travel pri...	en

5 rows × 24 columns

In [3]:
```
movie_data.shape
```

Out[3]:
```
(4803, 24)
```

In [4]:
```
#selecting the relevent feture
selecting_feature = ['genres','keywords','tagline','cast','director']
print(selecting_feature)
```

```
['genres', 'keywords', 'tagline', 'cast', 'director']
```

In [5]:
```
for feature in selecting_feature:
    movie_data[feature] = movie_data[feature].fillna('')
```

In [6]:
```
combine_feature = movie_data['genres']+' '+movie_data['keywords']+' '+movie_data['taglin
```

In [7]:
```
print(combine_feature)
```

```
0       Action Adventure Fantasy Science Fiction cultu...
1       Adventure Fantasy Action ocean drug abuse exot...
2       Action Adventure Crime spy based on novel secr...
3       Action Crime Drama Thriller dc comics crime fi...
4       Action Adventure Science Fiction based on nove...
                              ...
4798    Action Crime Thriller united states\u2013mexic...
4799    Comedy Romance  A newlywed couple's honeymoon ...
4800    Comedy Drama Romance TV Movie date love at fir...
4801      A New Yorker in Shanghai Daniel Henney Eliza...
4802    Documentary obsession camcorder crush dream gi...
Length: 4803, dtype: object
```

In [8]:
```
vectorizer = TfidfVectorizer()
```

In [26]:
```
feature_vector = vectorizer.fit_transform(combine_feature)
print(feature_vector)
```

```
  (0, 2432)     0.17272411194153
  (0, 7755)     0.1128035714854756
  (0, 13024)    0.1942362060108871
  (0, 10229)    0.16058685400095302
  (0, 8756)     0.22709015857011816
  (0, 14608)    0.15150672398763912
  (0, 16668)    0.19843263965100372
  (0, 14064)    0.20596090415084142
  (0, 13319)    0.2177470539412484
  (0, 17290)    0.20197912553916567
  (0, 17007)    0.23643326319898797
  (0, 13349)    0.15021264094167086
  (0, 11503)    0.27211310056983656
  (0, 11192)    0.09049319826481456
  (0, 16998)    0.1282126322850579
  (0, 15261)    0.07095833561276566
  (0, 4945)     0.24025852494110758
  (0, 14271)    0.21392179219912877
  (0, 3225)     0.24960162956997736
  (0, 16587)    0.12549432354918996
  (0, 14378)    0.33962752210959823
  (0, 5836)     0.1646750903586285
  (0, 3065)     0.22208377802661425
  (0, 3678)     0.21392179219912877
  (0, 5437)     0.1036413987316636
  :     :
  (4801, 17266) 0.2886098184932947
  (4801, 4835)  0.24713765026963996
  (4801, 403)   0.17727585190343226
  (4801, 6935)  0.2886098184932947
  (4801, 11663) 0.21557500762727902
  (4801, 1672)  0.1564793427630879
  (4801, 10929) 0.13504166990041588
  (4801, 7474)  0.11307961713172225
  (4801, 3796)  0.3342808988877418
  (4802, 6996)  0.5700048226105303
  (4802, 5367)  0.22969114490410403
  (4802, 3654)  0.262512960498006
  (4802, 2425)  0.24002350969074696
  (4802, 4608)  0.24002350969074696
  (4802, 6417)  0.21753405888348784
  (4802, 4371)  0.1538239182675544
  (4802, 12989) 0.1696476532191718
  (4802, 1316)  0.1960747079005741
  (4802, 4528)  0.19504460807622875
  (4802, 3436)  0.21753405888348784
  (4802, 6155)  0.18056463596934083
  (4802, 4980)  0.16078053641367315
  (4802, 2129)  0.3099656128577656
  (4802, 4518)  0.16784466610624255
  (4802, 11161) 0.17867407682173203
```

In [27]:
```
#this is use to find similarity between all movies.its mean how a movie is similar to ot
similarity = cosine_similarity(feature_vector)
```

In [28]:
```
print(similarity.shape)
```

```
(4803, 4803)
```

In [31]:
```
movie_name = input('Enter movie name:')
```

```
Enter movie name:iron man
```

In [32]:
```
#this list contain all the movie name that is in title.
title = movie_data['title'].tolist()
```

In [33]:
```
#finding close match between the movie name and list of all title
find_close_match = difflib.get_close_matches(movie_name, list_of_all_title)
print(find_close_match)
```

```
['Iron Man', 'Iron Man 3', 'Iron Man 2']
```

In [34]:
```
close_match = find_close_match[0]
print(close_match)
```

```
Iron Man
```

In [35]:
```
index_of_movie = movie_data[movie_data.title ==close_match]['index'].values[0]
print(index_of_movie)
```

```
68
```

In [36]:
```
#this is use to find the similarity of movie we given to all movies in title
similarity_score = list(enumerate(similarity[index_of_movie]))
```

In [37]:
```
#sorting the movie similarity
sorting_similarity = sorted(similarity_score, key = lambda x:x[1],reverse = True)
```

In [38]:
```
#printing the movies based on index
print('movie suggested for you :\n')
i=1
for movie in sorting_similarity:
    index =movie[0]
    title_from_index = movie_data[movie_data.index==index]['title'].values[0]
    if(i<20):
        print(i,',',title_from_index)
        i+=1
```

```
movie suggested for you :

1 , Iron Man
2 , Iron Man 2
3 , Iron Man 3
4 , Avengers: Age of Ultron
5 , The Avengers
6 , Captain America: Civil War
7 , Captain America: The Winter Soldier
8 , Ant-Man
9 , X-Men
10 , Made
11 , X-Men: Apocalypse
12 , X2
13 , The Incredible Hulk
14 , The Helix... Loaded
15 , X-Men: First Class
16 , X-Men: Days of Future Past
17 , Captain America: The First Avenger
18 , Kick-Ass 2
19 , Guardians of the Galaxy
```

In [39]:
```
#final of Movie Recommendation System
#this is the all step we have done before.
movie_name = input('Enter movie name:')

list_of_all_title = movie_data['title'].tolist()

find_close_match = difflib.get_close_matches(movie_name, list_of_all_title)
close_match = find_close_match[0]

e = movie_data[movie_data.title ==close_match]['index'].values[0]
```

```
similarity_score = list(enumerate(similarity[index_of_movie]))
sorting_similarity = sorted(similarity_score, key = lambda x:x[1],reverse = True)

print('movie suggested for you :\n')
i=1
for movie in sorting_similarity:
    index =movie[0]
    title_from_index = movie_data[movie_data.index==index]['title'].values[0]
    if(i<20):
        print(i,',',title_from_index)
        i+=1
```

```
Enter movie name:batman
movie suggested for you :

1 , Batman
2 , Batman Returns
3 , Batman & Robin
4 , The Dark Knight Rises
5 , Batman Begins
6 , The Dark Knight
7 , A History of Violence
8 , Superman
9 , Beetlejuice
10 , Bedazzled
11 , Mars Attacks!
12 , The Sentinel
13 , Planet of the Apes
14 , Man of Steel
15 , Suicide Squad
16 , The Mask
17 , Salton Sea
18 , Spider-Man 3
19 , The Postman Always Rings Twice
```

In [ ]: