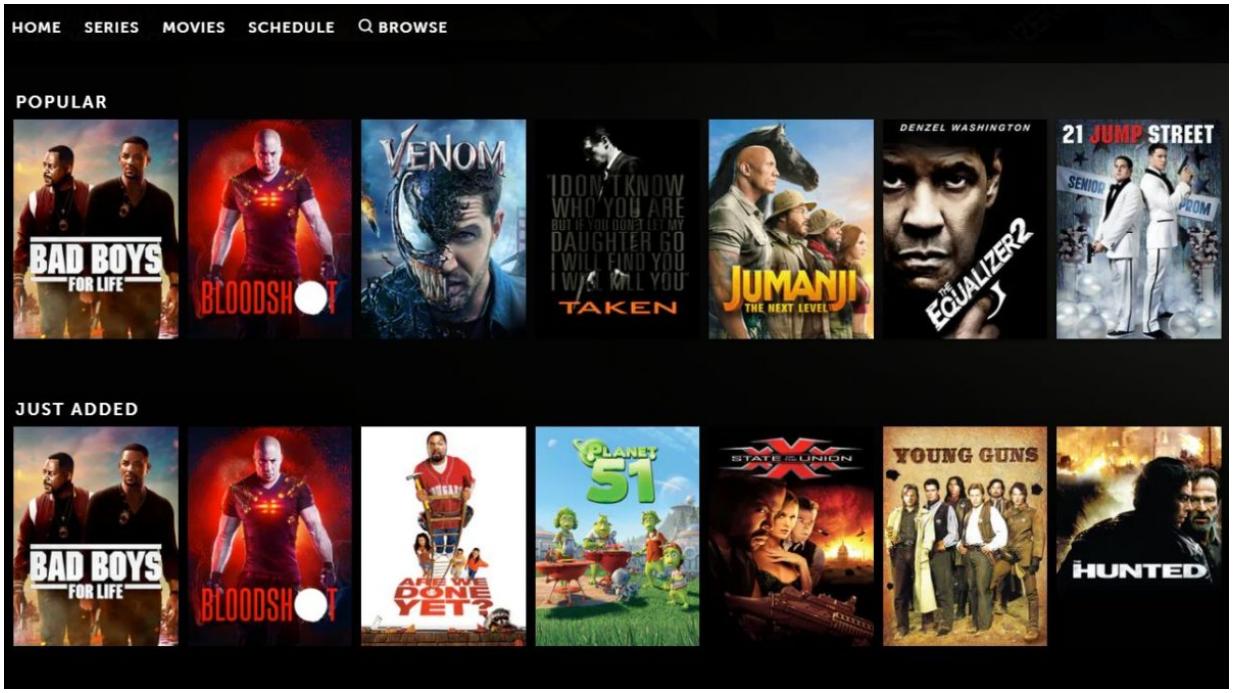
Content Based Recommender System



Outline

1. **4803 datasize** from www.kaggle.com (http://www.kaggle.com)
2. **Content Based** Recommender System
3. **Data Cleaning**
4. Model Bulding by **CountVectorizer** from sklearn.feature_extraction.text
5. Exporting the model into **pickle** file for production
6. Building a **Website** bu using **streamlit**

Recommender systems are machine learning systems that help users discover new product and services. Every time you shop online, a recommendation system is guiding you towards the most likely product you might purchase.

Types of Recommender Systems:

1. Content based: recommends based on similarity of content(tags)
2. Collaborative Filtering based: Collaborative filtering is based on the assumption that people who agreed in the past will agree in the future and that they will like similar kind of objects as they liked in the past. (user interest, like, share)
3. Hybrid: Content based + Collaborative Filtering

In [1]:
```
import numpy as np
import pandas as pd
```

here is the link for data (https://www.kaggle.com/tmdb/tmdb-movie-metadata)

In [2]:
```
movies = pd.read_csv('tmdb_5000_movies.csv')
credits = pd.read_csv('tmdb_5000_credits.csv')
```

In [3]:
```
movies.head(3)
```

Out[3]:

	budget	genres	homepage	id	keywords	original_
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, ...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, ...	

In [4]:
```
movies.shape
```

Out[4]:
```
(4803, 20)
```

In [5]:
```
movies.shape
```

Out[5]:
```
(4803, 20)
```

In [5]:
```
credits.head(3)
```

Out[5]:

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...

In [6]:
```
credits.head(2)['cast']
```

Out[6]:
```
0    [{"cast_id": 242, "character": "Jake Sully", "...
1    [{"cast_id": 4, "character": "Captain Jack Spa...
Name: cast, dtype: object
```

Merge Movies & Credits

In [4]:
```
movies = movies.merge(credits, on = 'title')
```

In [5]:
```
movies.shape
```

Out[5]:
```
(4809, 23)
```

In [6]:
```
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
```

In [9]:
```
movies['original_language'].value_counts()
```

Out[9]:
```
en    4510
fr      70
es      32
de      27
zh      27
hi      19
ja      16
it      14
cn      12
ko      12
ru      11
pt       9
da       7
sv       5
fa       4
nl       4
th       3
he       3
ta       2
cs       2
ro       2
ar       2
id       2
pl       1
af       1
tr       1
sl       1
vi       1
is       1
no       1
xx       1
nb       1
hu       1
te       1
el       1
ps       1
ky       1
Name: original_language, dtype: int64
```

In [10]:
```
movies.columns
# budget
# homepage
# id
# original_language
# original_title
# popularity
# production_comapny
# production_countries
# release-date(not sure)
```

Out[10]:
```
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')
```

In [11]:
```
#Keep these for data processing
df1 = movies[['movie_id','title','overview','genres','keywords','cast','crew']]
```

In [12]:
```
df1.head()
```

Out[12]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 1463, "name": "culture clash"}, {"id":...	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id" "52fe48009251416c750aca23' "de..
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id" "52fe4232c3a36847f800b579' "de..
2	206647	Spectre	A cryptic message from Bond's past sends him o...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 470, "name": "spy"}, {"id": 818, "name...	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id" "54805967c3a36829b5002c41' "de..
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	[{"id": 849, "name": "dc comics"}, {"id": 853,...	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id" "52fe4781c3a36847f81398c3' "de..
4	49529	John Carter	John Carter is a war-weary, former military ca...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 818, "name": "based on novel"}, {"id":...	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id" "52fe479ac3a36847f813eaa3' "de..

In [13]:
```
df1.isnull().sum()
```

Out[13]:
```
movie_id    0
title       0
overview    3
genres      0
keywords    0
cast        0
crew        0
dtype: int64
```

In [14]:
```
# drop the overview because of missing value
df1.dropna(inplace=True)
```

```
<ipython-input-14-9b46f6d1e071>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1.dropna(inplace=True)
```

In [15]:
```
df1.isnull().sum()
```

Out[15]:
```
movie_id    0
title       0
overview    0
genres      0
keywords    0
cast        0
crew        0
dtype: int64
```

In [16]:
```
df1.duplicated().sum()
```

Out[16]:
```
0
```

Pandas duplicated() method helps in analyzing duplicate values only. It returns a boolean series which is True only for Unique elements.

In [17]:
```
df1.head()
```

Out[17]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 1463, "name": "culture clash"}, {"id":...	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id" "52fe48009251416c750aca23' "de..
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id" "52fe4232c3a36847f800b579' "de..
2	206647	Spectre	A cryptic message from Bond's past sends him o...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 470, "name": "spy"}, {"id": 818, "name...	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id" "54805967c3a36829b5002c41' "de..
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	[{"id": 849, "name": "dc comics"}, {"id": 853,...	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id" "52fe4781c3a36847f81398c3' "de..
4	49529	John Carter	John Carter is a war-weary, former military ca...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 818, "name": "based on novel"}, {"id":...	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id" "52fe479ac3a36847f813eaa3' "de..

In [18]:
```
df1.iloc[0].genres
```

Out[18]:
```
'[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "na
me": "Fantasy"}, {"id": 878, "name": "Science Fiction"}]'
```

In [44]:
```
# '[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "nc
# ['Acton','Adventure','Fantasy', 'SciFi']
```

In [19]:
```
import ast
ast.literal_eval('[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"},
)
```

Out[19]:
```
[{'id': 28, 'name': 'Action'},
 {'id': 12, 'name': 'Adventure'},
 {'id': 14, 'name': 'Fantasy'},
 {'id': 878, 'name': 'Science Fiction'}]
```

In [20]:
```
def convert(text):
    L = []
    for i in ast.literal_eval(text):
        L.append(i['name'])
    return L
```

In [21]:
```
df1['genres'] = df1['genres'].apply(convert)
df1.head()
```

```
<ipython-input-21-852bdda8ca81>: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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['genres'] = df1['genres'].apply(convert)
```

Out[21]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[{"id": 1463, "name": "culture clash"}, {"id":...	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	A cryptic message from Bond’s past sends him o...	[Action, Adventure, Crime]	[{"id": 470, "name": "spy"}, {"id": 818, "name...	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[Action, Crime, Drama, Thriller]	[{"id": 849, "name": "dc comics"}, {"id": 853,...	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	John Carter is a war-weary, former military ca...	[Action, Adventure, Science Fiction]	[{"id": 818, "name": "based on novel"}, {"id":...	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

In [22]:
```
# Do the same thing for keywords
df1['keywords'] = df1['keywords'].apply(convert)
df1.head()
```

```
<ipython-input-22-ed43a6ff42f2>:2: 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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['keywords'] = df1['keywords'].apply(convert)
```

Out[22]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	A cryptic message from Bond's past sends him o...	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, mi...	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret i...	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	John Carter is a war-weary, former military ca...	[Action, Adventure, Science Fiction]	[based on novel, mars, medallion, space travel...	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

In [23]:
```
df1['cast'][0]
```

```
"name": "Sean Patrick Murphy", "order": 17}, {"cast_id": 38, "character": "Sh
uttle Crew Chief", "credit_id": "52fe48009251416c750aca73", "gender": 2, "i
d": 1019578, "name": "Peter Dillon", "order": 18}, {"cast_id": 39, "characte
r": "Tractor Operator / Troupe", "credit_id": "52fe48009251416c750aca77", "ge
nder": 0, "id": 91443, "name": "Kevin Dorman", "order": 19}, {"cast_id": 40,
"character": "Dragon Gunship Pilot", "credit_id": "52fe48009251416c750aca7b",
"gender": 2, "id": 173391, "name": "Kelson Henderson", "order": 20}, {"cast_i
d": 41, "character": "Dragon Gunship Gunner", "credit_id": "52fe48009251416c7
50aca7f", "gender": 0, "id": 1207236, "name": "David Van Horn", "order": 21},
{"cast_id": 42, "character": "Dragon Gunship Navigator", "credit_id": "52fe48
009251416c750aca83", "gender": 0, "id": 215913, "name": "Jacob Tomuri", "orde
r": 22}, {"cast_id": 43, "character": "Suit #1", "credit_id": "52fe4800925141
6c750aca87", "gender": 0, "id": 143206, "name": "Michael Blain-Rozgay", "orde
r": 23}, {"cast_id": 44, "character": "Suit #2", "credit_id": "52fe4800925141
6c750aca8b", "gender": 2, "id": 169676, "name": "Jon Curry", "order": 24},
{"cast_id": 46, "character": "Ambient Room Tech", "credit_id": "52fe480092514
16c750aca8f", "gender": 0, "id": 1048610, "name": "Luke Hawker", "order": 2
5}, {"cast_id": 47, "character": "Ambient Room Tech / Troupe", "credit_id":
"52fe48009251416c750aca93", "gender": 0, "id": 42288, "name": "Woody Schult
```

In [24]:
```
def convert3(text):
    L = []
    counter = 0
    for i in ast.literal_eval(text):
        if counter < 3:
            L.append(i['name'])
        counter+=1
    return L
```

In [25]:
```
df1['cast'].apply(convert3)
```

Out[25]:
```
0        [Sam Worthington, Zoe Saldana, Sigourney Weaver]
1           [Johnny Depp, Orlando Bloom, Keira Knightley]
2            [Daniel Craig, Christoph Waltz, Léa Seydoux]
3            [Christian Bale, Michael Caine, Gary Oldman]
4          [Taylor Kitsch, Lynn Collins, Samantha Morton]
                              ...
4804    [Carlos Gallardo, Jaime de Hoyos, Peter Marqua...
4805         [Edward Burns, Kerry Bishé, Marsha Dietlein]
4806           [Eric Mabius, Kristin Booth, Crystal Lowe]
4807            [Daniel Henney, Eliza Coupe, Bill Paxton]
4808    [Drew Barrymore, Brian Herzlinger, Corey Feldman]
Name: cast, Length: 4806, dtype: object
```

In [40]:
```
#movies['cast'] = movies['cast'].apply(lambda x:x[0:3])
```

In [26]:
```
df1['cast'] = df1['cast'].apply(convert3)
df1.head()
```

```
<ipython-input-26-ba26654517a6>: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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['cast'] = df1['cast'].apply(convert3)
```

Out[26]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	[Sam Worthington, Zoe Saldana, Sigourney Weaver]	[{"credit_id" "52fe48009251416c750aca23" "de..
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...	[Johnny Depp, Orlando Bloom, Keira Knightley]	[{"credit_id" "52fe4232c3a36847f800b579" "de..
2	206647	Spectre	A cryptic message from Bond’s past sends him o...	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, mi...	[Daniel Craig, Christoph Waltz, Léa Seydoux]	[{"credit_id" "54805967c3a36829b5002c41" "de..
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret i...	[Christian Bale, Michael Caine, Gary Oldman]	[{"credit_id" "52fe4781c3a36847f81398c3" "de..
4	49529	John Carter	John Carter is a war-weary, former military ca...	[Action, Adventure, Science Fiction]	[based on novel, mars, medallion, space travel...	[Taylor Kitsch, Lynn Collins, Samantha Morton]	[{"credit_id" "52fe479ac3a36847f813eaa3" "de..

The first 3 of the cast is appeard.

In [27]:
```
df1['crew'][0]
```

Out[27]:
```
'[{"credit_id": "52fe48009251416c750aca23", "department": "Editing", "gende
r": 0, "id": 1721, "job": "Editor", "name": "Stephen E. Rivkin"}, {"credit_i
d": "539c47ecc3a36810e3001f87", "department": "Art", "gender": 2, "id": 496,
"job": "Production Design", "name": "Rick Carter"}, {"credit_id": "54491c89c3
a3680fb4001cf7", "department": "Sound", "gender": 0, "id": 900, "job": "Sound
Designer", "name": "Christopher Boyes"}, {"credit_id": "54491cb70e0a267480001
bd0", "department": "Sound", "gender": 0, "id": 900, "job": "Supervising Soun
d Editor", "name": "Christopher Boyes"}, {"credit_id": "539c4a4cc3a36810c9002
101", "department": "Production", "gender": 1, "id": 1262, "job": "Casting",
"name": "Mali Finn"}, {"credit_id": "5544ee3b925141499f0008fc", "department":
"Sound", "gender": 2, "id": 1729, "job": "Original Music Composer", "name":
"James Horner"}, {"credit_id": "52fe48009251416c750ac9c3", "department": "Dir
ecting", "gender": 2, "id": 2710, "job": "Director", "name": "James Camero
n"}, {"credit_id": "52fe48009251416c750ac9d9", "department": "Writing", "gend
er": 2, "id": 2710, "job": "Writer", "name": "James Cameron"}, {"credit_id":
"52fe48009251416c750aca17", "department": "Editing", "gender": 2, "id": 2710,
"job": "Editor", "name": "James Cameron"}, {"credit_id": "52fe48009251416c750
aca29", "department": "Production", "gender": 2, "id": 2710, "job": "Produce
r", "name": "James Cameron"}, {"credit_id": "52fe48009251416c750aca3f", "depa
```

In [28]:
```
# We just care about director from the crew
def fetch_director(text):
    L = []
    for i in ast.literal_eval(text):
        if i['job'] == 'Director':
            L.append(i['name'])
    return L
```

In [29]:
```
df1['crew'].apply(fetch_director)
```

Out[29]:
```
0                                  [James Cameron]
1                                [Gore Verbinski]
2                                     [Sam Mendes]
3                              [Christopher Nolan]
4                                 [Andrew Stanton]
                              ...
4804                             [Robert Rodriguez]
4805                                 [Edward Burns]
4806                                  [Scott Smith]
4807                                  [Daniel Hsia]
4808    [Brian Herzlinger, Jon Gunn, Brett Winn]
Name: crew, Length: 4806, dtype: object
```

In [30]:
```
df1['crew'] = df1['crew'].apply(fetch_director)
df1.head(5)
```

```
<ipython-input-30-102b92940dd2>: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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['crew'] = df1['crew'].apply(fetch_director)
```

Out[30]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	[Sam Worthington, Zoe Saldana, Sigourney Weaver]	[James Cameron]
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...	[Johnny Depp, Orlando Bloom, Keira Knightley]	[Gore Verbinski]
2	206647	Spectre	A cryptic message from Bond's past sends him o...	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, mi...	[Daniel Craig, Christoph Waltz, Léa Seydoux]	[Sam Mendes]
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret i...	[Christian Bale, Michael Caine, Gary Oldman]	[Christopher Nolan]
4	49529	John Carter	John Carter is a war-weary, former military ca...	[Action, Adventure, Science Fiction]	[based on novel, mars, medallion, space travel...	[Taylor Kitsch, Lynn Collins, Samantha Morton]	[Andrew Stanton]

In [31]:
```
df1['overview'][0]
```

Out[31]:
```
'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.'
```

In [32]:
```
df1['overview'].apply(lambda x:x.split())
```

Out[32]:
```
0       [In, the, 22nd, century,, a, paraplegic, Marin...
1       [Captain, Barbossa,, long, believed, to, be, d...
2       [A, cryptic, message, from, Bond’s, past, send...
3       [Following, the, death, of, District, Attorney...
4       [John, Carter, is, a, war-weary,, former, mili...
                              ...
4804    [El, Mariachi, just, wants, to, play, his, gui...
4805    [A, newlywed, couple's, honeymoon, is, upended...
4806    ["Signed,, Sealed,, Delivered", introduces, a,...
4807    [When, ambitious, New, York, attorney, Sam, is...
4808    [Ever, since, the, second, grade, when, he, fi...
Name: overview, Length: 4806, dtype: object
```

In [33]:
```
df1['overview'] = df1['overview'].apply(lambda x:x.split())
df1.head(5)
```

```
<ipython-input-33-da04b682a57e>: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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['overview'] = df1['overview'].apply(lambda x:x.split())
```

Out[33]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	[In, the, 22nd, century,, a, paraplegic, Marin...	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	[Sam Worthington, Zoe Saldana, Sigourney Weaver]	[James Cameron]
1	285	Pirates of the Caribbean: At World's End	[Captain, Barbossa,, long, believed, to, be, d...	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...	[Johnny Depp, Orlando Bloom, Keira Knightley]	[Gore Verbinski]
2	206647	Spectre	[A, cryptic, message, from, Bond’s, past, send...	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, mi...	[Daniel Craig, Christoph Waltz, Léa Seydoux]	[Sam Mendes]
3	49026	The Dark Knight Rises	[Following, the, death, of, District, Attorney...	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret i...	[Christian Bale, Michael Caine, Gary Oldman]	[Christopher Nolan]
4	49529	John Carter	[John, Carter, is, a, war-weary,, former, mili...	[Action, Adventure, Science Fiction]	[based on novel, mars, medallion, space travel...	[Taylor Kitsch, Lynn Collins, Samantha Morton]	[Andrew Stanton]

Sam Worthington ==> SamWorthington. The space between all the text data must be omitted because it considered as a two person. So we should apply a new function to

remove space between genres, cast, crew, and keywords.

In [34]:
```
df1['genres'].apply(lambda x:[i.replace(" ","") for i in x])
```

Out[34]:
```
0       [Action, Adventure, Fantasy, ScienceFiction]
1                       [Adventure, Fantasy, Action]
2                         [Action, Adventure, Crime]
3                    [Action, Crime, Drama, Thriller]
4                 [Action, Adventure, ScienceFiction]
                              ...
4804                       [Action, Crime, Thriller]
4805                               [Comedy, Romance]
4806                [Comedy, Drama, Romance, TVMovie]
4807                                               []
4808                                    [Documentary]
Name: genres, Length: 4806, dtype: object
```

In [35]:
```
df1['genres']= df1['genres'].apply(lambda x:[i.replace(" ","") for i in x])
df1['cast']= df1['cast'].apply(lambda x:[i.replace(" ","") for i in x])
df1['crew']= df1['crew'].apply(lambda x:[i.replace(" ","") for i in x])
df1['keywords']= df1['keywords'].apply(lambda x:[i.replace(" ","") for i in x])
```

```
<ipython-input-35-05f2b059f4e0>: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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['genres']= df1['genres'].apply(lambda x:[i.replace(" ","") for i in x])
<ipython-input-35-05f2b059f4e0>:2: 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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['cast']= df1['cast'].apply(lambda x:[i.replace(" ","") for i in x])
<ipython-input-35-05f2b059f4e0>:3: 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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['crew']= df1['crew'].apply(lambda x:[i.replace(" ","") for i in x])
<ipython-input-35-05f2b059f4e0>:4: 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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['keywords']= df1['keywords'].apply(lambda x:[i.replace(" ","") for i in
 x])
```

In [36]:
```
df1.head(5)
```

Out[36]:

	movie_id	title	overview	genres	keywords	cast	cre
0	19995	Avatar	[In, the, 22nd, century,, a, paraplegic, Marin...	[Action, Adventure, Fantasy, ScienceFiction]	[cultureclash, future, spacewar, spacecolony, ...	[SamWorthington, ZoeSaldana, SigourneyWeaver]	[JamesCamerc
1	285	Pirates of the Caribbean: At World's End	[Captain, Barbossa,, long, believed, to, be, d...	[Adventure, Fantasy, Action]	[ocean, drugabuse, exoticisland, eastindiatrad...	[JohnnyDepp, OrlandoBloom, KeiraKnightley]	[GoreVerbins
2	206647	Spectre	[A, cryptic, message, from, Bond's, past, send...	[Action, Adventure, Crime]	[spy, basedonnovel, secretagent, sequel, mi6, ...	[DanielCraig, ChristophWaltz, LéaSeydoux]	[SamMende
3	49026	The Dark Knight Rises	[Following, the, death, of, District, Attorney...	[Action, Crime, Drama, Thriller]	[dccomics, crimefighter, terrorist, secretiden...	[ChristianBale, MichaelCaine, GaryOldman]	[ChristopherNola
4	49529	John Carter	[John, Carter, is, a, war-weary,, former, mili...	[Action, Adventure, ScienceFiction]	[basedonnovel, mars, medallion, spacetravel, p...	[TaylorKitsch, LynnCollins, SamanthaMorton]	[AndrewStantc

Creating tags

In [37]:
```
df1['tags'] = df1['overview'] + df1['genres'] + df1['keywords'] + df1['cast'] + d
```

```
<ipython-input-37-56936bc97bca>: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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['tags'] = df1['overview'] + df1['genres'] + df1['keywords'] + df1['cast']
+ df1['crew']
```

In [38]:
```
df1.head(5)
```

Out[38]:

	movie_id	title	overview	genres	keywords	cast	
0	19995	Avatar	[In, the, 22nd, century,, a, paraplegic, Marin...	[Action, Adventure, Fantasy, ScienceFiction]	[cultureclash, future, spacewar, spacecolony, ...	[SamWorthington, ZoeSaldana, SigourneyWeaver]	[JamesCam
1	285	Pirates of the Caribbean: At World's End	[Captain, Barbossa,, long, believed, to, be, d...	[Adventure, Fantasy, Action]	[ocean, drugabuse, exoticisland, eastindiatrad...	[JohnnyDepp, OrlandoBloom, KeiraKnightley]	[GoreVerb
2	206647	Spectre	[A, cryptic, message, from, Bond's, past, send...	[Action, Adventure, Crime]	[spy, basedonnovel, secretagent, sequel, mi6, ...	[DanielCraig, ChristophWaltz, LéaSeydoux]	[SamMe
3	49026	The Dark Knight Rises	[Following, the, death, of, District, Attorney...	[Action, Crime, Drama, Thriller]	[dccomics, crimefighter, terrorist, secretiden...	[ChristianBale, MichaelCaine, GaryOldman]	[ChristopherN
4	49529	John Carter	[John, Carter, is, a, war-weary,, former, mili...	[Action, Adventure, ScienceFiction]	[basedonnovel, mars, medallion, spacetravel, p...	[TaylorKitsch, LynnCollins, SamanthaMorton]	[AndrewSta

We just need movie_id, title, and tags

In [39]:
```
new_df = df1[['movie_id','title','tags']]
new_df.head(5)
```

Out[39]:

	movie_id	title	tags
0	19995	Avatar	[In, the, 22nd, century,, a, paraplegic, Marin...
1	285	Pirates of the Caribbean: At World's End	[Captain, Barbossa,, long, believed, to, be, d...
2	206647	Spectre	[A, cryptic, message, from, Bond's, past, send...
3	49026	The Dark Knight Rises	[Following, the, death, of, District, Attorney...
4	49529	John Carter	[John, Carter, is, a, war-weary,, former, mili...

Now we want to add space between tags

In [40]:
```
new_df['tags'].apply(lambda x: " ".join(x))
```

Out[40]:
```
0       In the 22nd century, a paraplegic Marine is di...
1       Captain Barbossa, long believed to be dead, ha...
2       A cryptic message from Bond’s past sends him o...
3       Following the death of District Attorney Harve...
4       John Carter is a war-weary, former military ca...
                              ...
4804    El Mariachi just wants to play his guitar and ...
4805    A newlywed couple's honeymoon is upended by th...
4806    "Signed, Sealed, Delivered" introduces a dedic...
4807    When ambitious New York attorney Sam is sent t...
4808    Ever since the second grade when he first saw ...
Name: tags, Length: 4806, dtype: object
```

In [41]:
```
new_df['tags'] = new_df['tags'].apply(lambda x: " ".join(x))
new_df.head(3)
```

```
<ipython-input-41-cf22c00ff523>: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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  new_df['tags'] = new_df['tags'].apply(lambda x: " ".join(x))
```

Out[41]:

	movie_id	title	tags
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...

In [42]:
```
new_df['tags'][0]
```

Out[42]:
```
'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. 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 SamWorthington ZoeSaldana SigourneyWeaver JamesCameron'
```

Everything is in the above information for searching a movie.

In [43]:
```
# It is batter for tags to change the words into lower case
new_df['tags'] = new_df['tags'].apply(lambda x:x.lower())
new_df.head(5)
```

```
<ipython-input-43-dd6796bee4c8>:2: 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-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  new_df['tags'] = new_df['tags'].apply(lambda x:x.lower())
```

Out[43]:

	movie_id	title	tags
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...

In [44]:
```
new_df['tags'][0]
```

Out[44]:
```
'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. 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 samworthington zoesaldana sigourneyweaver jamescameron'
```

In [45]:
```
new_df['tags'][1]
```

Out[45]:
```
"captain barbossa, long believed to be dead, has come back to life and is headed to the edge of the earth with will turner and elizabeth swann. but nothing is quite as it seems. adventure fantasy action ocean drugabuse exoticisland eastindiatradingcompany loveofone'slife traitor shipwreck strongwoman ship alliance calypso afterlife fighter pirate swashbuckler aftercreditsstinger johnnydepp orlandobloom keiraknightley goreverbinski"
```

Here we are dealing with text data instead of numerical one. First we have to convert each tags text data into vector (text vectorization). Then we have to calculate the most common tags for each movie. So we use Bags of Words. Depending on the problem you should choose one method. Here we have to calculate the most frequent tags for each movie, and as a result we use Bag of Words Term Frequency.

Text Vectorization

Text Vectorization is the process of converting text into numerical representation. Here is some popular methods to accomplish text vectorization:

1. Binary Term Frequency
2. Bag of Words (BoW) Term Frequency
3. (L1) Normalized Term Frequency
4. (L2) Normalized TF-IDF
5. Word2Vec

Model Building

In [46]:
```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=5000,stop_words='english')
```

In [47]:
```
# convert tags into array
cv.fit_transform(new_df['tags']).toarray()
```

Out[47]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

In [48]:
```
vector = cv.fit_transform(new_df['tags']).toarray()
```

In [49]:
```
vector.shape
```

Out[49]:
```
(4806, 5000)
```

In [57]:
```
vector[0]
```

Out[57]:
```
array([0, 0, 0, ..., 0, 0, 0], dtype=int64)
```

This is the first movie.

In [50]:
```
from sklearn.metrics.pairwise import cosine_similarity
similarity = cosine_similarity(vector)
```

In [51]:
```
similarity
```

Out[51]:
```
array([[1.        , 0.0860309 , 0.05735393, ..., 0.0244558 , 0.0270369 ,
        0.        ],
       [0.0860309 , 1.        , 0.0625    , ..., 0.02665009, 0.        ,
        0.        ],
       [0.05735393, 0.0625    , 1.        , ..., 0.02665009, 0.        ,
        0.        ],
       ...,
       [0.0244558 , 0.02665009, 0.02665009, ..., 1.        , 0.07537784,
        0.0489116 ],
       [0.0270369 , 0.        , 0.        , ..., 0.07537784, 1.        ,
        0.05407381],
       [0.        , 0.        , 0.        , ..., 0.0489116 , 0.05407381,
        1.        ]])
```

In [52]:
```
similarity.shape
```

Out[52]:
```
(4806, 4806)
```

similrity matrix is an array of arrays. It is a diogonal Matrix.

In [53]:
```
similarity[0]
```

Out[53]:
```
array([1.        , 0.0860309 , 0.05735393, ..., 0.0244558 , 0.0270369 ,
       0.        ])
```

In [52]:
```
new_df[new_df['title'] == 'Avatar']
```

Out[52]:

	movie_id	title	tags
0	19995	Avatar	in the 22nd century, a paraplegic marine is di...

In [55]:
```
new_df[new_df['title'] == 'Avatar'].index
```

Out[55]:
```
Int64Index([0], dtype='int64')
```

In [56]:
```
new_df[new_df['title'] == 'Batman Begins'].index
```

Out[56]:
```
Int64Index([119], dtype='int64')
```

In [68]:
```
sorted(similarity[0],reverse=True)[0:5]
```

Out[68]:
```
[1.0000000000000002,
 0.25038669783359574,
 0.2421000623531261,
 0.24061325159289396,
 0.23939494881986934]
```

In [70]:
```
list(enumerate(similarity[0]))
```

Out[70]:
```
[(0, 1.0000000000000002),
 (1, 0.08603090020146065),
 (2, 0.057353933467640436),
 (3, 0.03823595564509363),
 (4, 0.177343107178349),
 (5, 0.11357771260606365),
 (6, 0.022282825891079324),
 (7, 0.1692777916923361),
 (8, 0.06131393394849658),
 (9, 0.07421560439929402),
 (10, 0.11295649894498103),
 (11, 0.07792865001991967),
 (12, 0.09197090092274487),
 (13, 0.04543108504242546),
 (14, 0.11128297681493143),
 (15, 0.04947706959952935),
 (16, 0.07894736842105264),
 (17, 0.1442149876003076),
 (18, 0.10838874619051501),
```

Created the list of tuples. first movie distance 0 is 1 and ... .

In [53]:
```
sorted(list(enumerate(similarity[0])),reverse=True,key=lambda x:x[1])[1:6]
```

Out[53]:
```
[(539, 0.25038669783359574),
 (1194, 0.2421000623531261),
 (507, 0.24061325159289396),
 (1216, 0.23939494881986934),
 (260, 0.23693955110363693)]
```

With the code above, sorted from maximum to minimum. the first one is match with the movie.

Be careful: The recommended movie is not that movie, it is the first 5 nearest distance to the movie.

In [54]:
```
def recommend(movie):
    movie_index = new_df[new_df['title'] == movie].index[0]
    distances = sorted(list(enumerate(similarity[movie_index])),reverse=True,key
    for i in distances:
        print(new_df.iloc[i[0]].title)
```

In [55]:
```
recommend('Batman Begins')
```

```
The Dark Knight
The Dark Knight Rises
Batman
Batman & Robin
Batman
```

Export the model to a pickle file

In [56]:
```
import pickle
```

In [58]:
```
# Rather than transfering a dataframe to that website you should transfer dictior
pickle.dump(new_df.to_dict(),open('movie_dict.pkl','wb'))
pickle.dump(similarity,open('similarity.pkl','wb'))
```

C:\Users\name of your PC\movie_dict.pkl & similarity.pkl => copy & paste these files into your app folder.

Date	Author
12-12-2021	**Ehsan Zia**