```
import numpy as np
import pandas as pd


movies=pd.read_csv("/content/drive/MyDrive/ML data/Movie/movies.csv")
credits=pd.read_csv("/content/drive/MyDrive/ML data/Movie/credits.csv")


movies.shape


movies.head(1)



credits.shape


credits.head(1)


database=movies
database.shape
database=database.merge(credits,left_on='id',right_on='movie_id')


database.shape


database.head(1)


database.info()


database=database[['id','title_x','genres','overview','keywords','cast','crew','runtime','original_language']]
#filtering data


database.head(1)


database.rename(columns={'title_x':'title','original_language':'language'},inplace=True)


database.head(1)


database.isnull().sum()


database.dropna(inplace=True)


database.isnull().sum()


import json


def merge(stringjson):
    l=[]
  #res=json.loads(database.head(1)['genres'].values[0])
    res=json.loads(stringjson)
    for i in res:
      l.append(i["name"])
    #print(l)
    return l



database['genres']=database['genres'].apply(merge)
database['keywords']=database['keywords'].apply(merge)


database['language'].isnull().sum()

     0


database.head(5)
```

	id	title	genres	overview	keywords	cast	crew	runtime	l
0	19995	Avatar	[Action, Adventure, Fantasy, Science Fiction]	In the 22nd century, a paraplegic Marine is di...	[culture clash, future, space war, space colon...	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...	162.0	
1	285	Pirates of the Caribbean: At World's End	[Adventure, Fantasy, Action]	Captain Barbossa, long believed to be dead, ha...	[ocean, drug abuse, exotic island, east india ...	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...	169.0	
				A cryptic	[spy,	[{"cast_id":			

```
import json


def filtercast(stringjson):
    l=[]
    cnt=0
  #res=json.loads(database.head(1)['genres'].values[0])
    res=json.loads(stringjson)
    for i in res:
      l.append(i["name"])
      cnt+=1
      if(cnt==3):
        break
    return l


database['cast']=database['cast'].apply(filtercast)


import json


def filtercrew(stringjson):
    l=[]
    cnt=0
  #res=json.loads(database.head(1)['genres'].values[0])
    res=json.loads(stringjson)
    for i in res:
      if(i['job']=='Director') :
        l.append(i["name"])
        cnt+=1
      elif(i['job']=='Producer') :
        l.append(i["name"])
        cnt+=1
      elif(i['job']=='Writer') :
        l.append(i["name"])
        cnt+=1
      if(cnt==3):
                break
    return l


database['crew']=database['crew'].apply(filtercrew)


database['overview']=database['overview'].apply(lambda x:[i.replace(" ","") for i in x])
database['genres']=database['genres'].apply(lambda x:[i.replace(" ","") for i in x])
database['crew']=database['crew'].apply(lambda x:[i.replace(" ","") for i in x])
database['cast']=database['cast'].apply(lambda x:[i.replace(" ","") for i in x])
database['keywords']=database['keywords'].apply(lambda x:[i.replace(" ","") for i in x])



import math

database['runtime']=database['runtime'].apply(lambda x:[str(math.ceil(x/30))])


database['language']=database['language'].apply(lambda x: [x])
```

```
database['tags']=database['genres']+database['overview']+database['keywords']+database['cast']+database['crew']+database['runtime']+database[
```

```
database.head(2)
```

	id	title	genres	overview	keywords	cast	crew	runtime	]
0	19995	Avatar	[Action, Adventure, Fantasy, ScienceFiction]	[In, the, 22nd, century,, a, paraplegic, Marin...	[cultureclash, future, spacewar, spacecolony, ...	[SamWorthington, ZoeSaldana, SigourneyWeaver]	[JamesCameron, JamesCameron, JamesCameron]	[6]	
1	285	Pirates of the Caribbean:	[Adventure, Fantasy,	[Captain, Barbossa,, long,	[ocean, drugabuse, exoticisland	[JohnnyDepp, OrlandoBloom,	[GoreVerbinski, JerryBruckheimer,	[6]	

```
model_data=database[["id","title","tags"]]
```

```
def modlify(l):
  s=""
  for i in range(len(l)-1):
    s+=l[i].lower()+" "
  s+=l[len(l)-1].lower()
  return s

model_data['tags']=model_data['tags'].apply(modlify)
```

```
model_data.head(2)
```

```
!pip install nltk
from nltk.stem.porter import PorterStemmer
ps=PorterStemmer()
def stem(text):
  y=[]
  for i in text.split():
    y.append(ps.stem(i))
  return " ".join(y)
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.6)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.3.1)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2022.10.31)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.65.0)
```

```
model_data['tags']=model_data['tags'].apply(stem)
```

```
<ipython-input-20-d517b83ae2fd>: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
  model_data['tags']=model_data['tags'].apply(stem)
```

```
!pip install scikit-learn
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer(max_features=5000,stop_words='english')
```

```
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.22.4)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.10.1)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.3.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.2.0)
```

```
vectors=cv.fit_transform(model_data['tags']).toarray()
```

```
vectors[0]
```

```
array([0, 0, 0, ..., 0, 0, 0])
```

```
for i in cv.get_feature_names_out():
  print(i,end=" ")


stem("louvre loved loving")


from sklearn.metrics.pairwise import cosine_similarity
similarity=cosine_similarity(vectors)


def recommend(movie):
  if(len(model_data[model_data['title']==movie])):
    movie_index=model_data[model_data['title']==movie].index[0]
    distances=similarity[movie_index]
    movies_list = sorted(list(enumerate(distances)),reverse=True,key=lambda x:x[1])[1:6]
    for i in movies_list:
      print(model_data.iloc[i[0]].title)
  else:
    print("Movie Unavailable in the Database")


import pickle
pickle.dump(similarity,open('similarity.pkl','wb'))


recommend('Joker')


#!pip install pickle
import pickle
pickle.dump(model_data,open('movies.pkl','wb'))
```