PUNE INSTITUTE OF COMPUTER TECHNOLOGY

Department of Computer Engineering

(2021-2022)

DSBDAL

Movie recommendation model

Group members:

1. 31146: Sanket Kulkarni
2. 31147: Sourabh Kumbhar

Guided by: Prof. N. P. Jadhav

Problem Statement:

Develop a movie recommendation model using the scikit-learn library in python.

Objectives:

- To learn the working of scikit-learn library and all the related functions.
- To understand concepts such as Cosine similarity and Count-Vectorizer.
- To develop a model that judges similarities between entities on many factors.
- To analyze our model on the basis of techniques like similarity matrix.

Theory:

AI Recommendation System:

It is a model / engine that uses machine learning to predict the users' choices and offer relevant suggestions to users. It filters and recommends the most suitable options to the users, hence aiding in the selection process of the user.
From a business standpoint, it promotes better customer engagement, thus resulting in higher sales. It also provides the customer with insights into similar products, thus increasing one's field of vision while choosing a product.

Scikit-learn:

It is a free and open-source machine learning library for Python. It provides a variety of classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Difflib:

This module provides classes and functions for comparing sequences. It can be used for example, for comparing files, and can produce information about file differences in various formats, including HTML and context and unified diffs.

In this program, we use the get_close_matches() function that returns a list of matches that have a high accuracy to the search parameter.

CountVectorizer:

It converts a collection of text documents to a matrix of token counts. This implementation produces a sparse representation of the counts using scipy.sparse.csr_matrix.

If you do not provide an a-priori dictionary and you do not use an analyzer that does some kind of feature selection then the number of features will be equal to the vocabulary size found by analyzing the data.

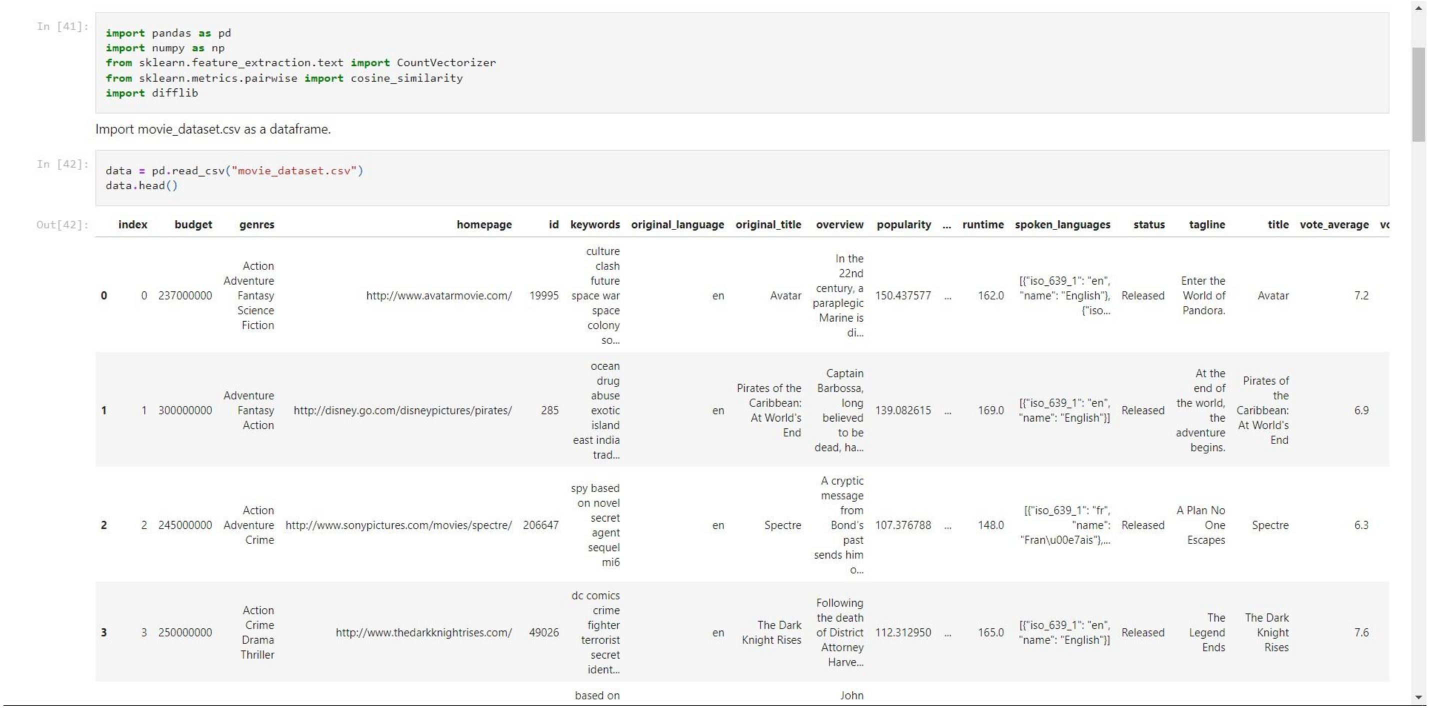
Cosine_Similarity:

It computes cosine similarity between samples in X and Y. Cosine similarity, or the cosine kernel, computes similarity as the normalized dot product of X and Y:

$K(X, Y) = <X, Y> / (||X||*||Y||)$

On L2-normalized data, this function is equivalent to linear_kernel.

Implementation:



In [41]:

```python
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import difflib
```

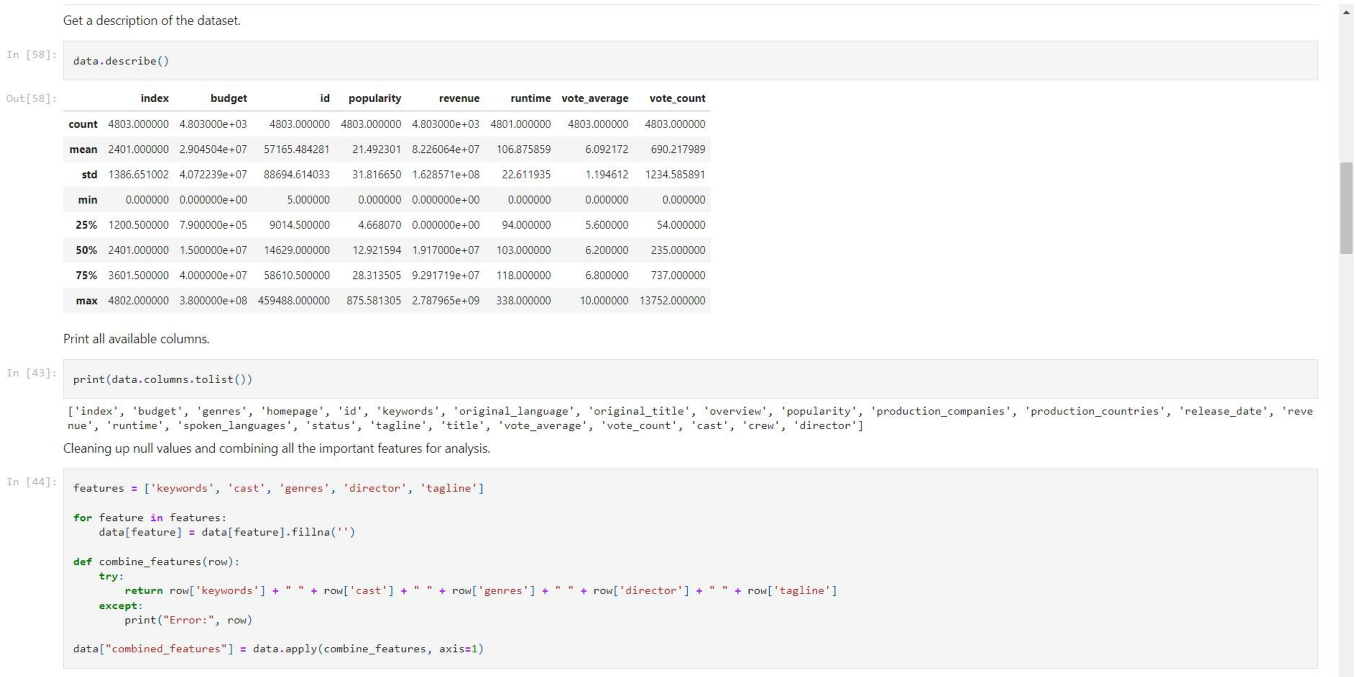
Import movie_dataset.csv as a dataframe.

In [42]:

```python
data = pd.read_csv("movie_dataset.csv")
data.head()
```

Out[42]:

	index	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	...	runtime	spoken_languages	status	tagline	title	vote_average	vc
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	19995	culture clash future space war space colony so...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	...	162.0	[{"iso_639_1": "en", "name": "English"}, {"iso...	Released	Enter the World of Pandora.	Avatar	7.2	
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285	ocean drug abuse exotic island east india trad...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	...	169.0	[{"iso_639_1": "en", "name": "English"}]	Released	At the end of the world, the adventure begins.	Pirates of the Caribbean: At World's End	6.9	
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647	spy based on novel secret agent sequel mi6	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	...	148.0	[{"iso_639_1": "fr", "name": "Fran\u00e7ais"},...	Released	A Plan No One Escapes	Spectre	6.3	
3	3	250000000	Action Crime Drama Thriller	http://www.thedarkknightrises.com/	49026	dc comics crime fighter terrorist secret ident...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950	...	165.0	[{"iso_639_1": "en", "name": "English"}]	Released	The Legend Ends	The Dark Knight Rises	7.6	
						based on			John									



Get a description of the dataset.

In [58]:

```python
data.describe()
```

Out[58]:

	index	budget	id	popularity	revenue	runtime	vote_average	vote_count
count	4803.000000	4.803000e+03	4803.000000	4803.000000	4.803000e+03	4801.000000	4803.000000	4803.000000
mean	2401.000000	2.904504e+07	57165.484281	21.492301	8.226064e+07	106.875859	6.092172	690.217989
std	1386.651002	4.072239e+07	88694.614033	31.816650	1.628571e+08	22.611935	1.194612	1234.585891
min	0.000000	0.000000e+00	5.000000	0.000000	0.000000e+00	0.000000	0.000000	0.000000
25%	1200.500000	7.900000e+05	9014.500000	4.668070	0.000000e+00	94.000000	5.600000	54.000000
50%	2401.000000	1.500000e+07	14629.000000	12.921594	1.917000e+07	103.000000	6.200000	235.000000
75%	3601.500000	4.000000e+07	58610.500000	28.313505	9.291719e+07	118.000000	6.800000	737.000000
max	4802.000000	3.800000e+08	459488.000000	875.581305	2.787965e+09	338.000000	10.000000	13752.000000

Print all available columns.

In [43]:

```python
print(data.columns.tolist())
```

```
['index', 'budget', 'genres', 'homepage', 'id', 'keywords', 'original_language', 'original_title', 'overview', 'popularity', 'production_companies', 'production_countries', 'release_date', 'reve
nue', 'runtime', 'spoken_languages', 'status', 'tagline', 'title', 'vote_average', 'vote_count', 'cast', 'crew', 'director']
```

Cleaning up null values and combining all the important features for analysis.

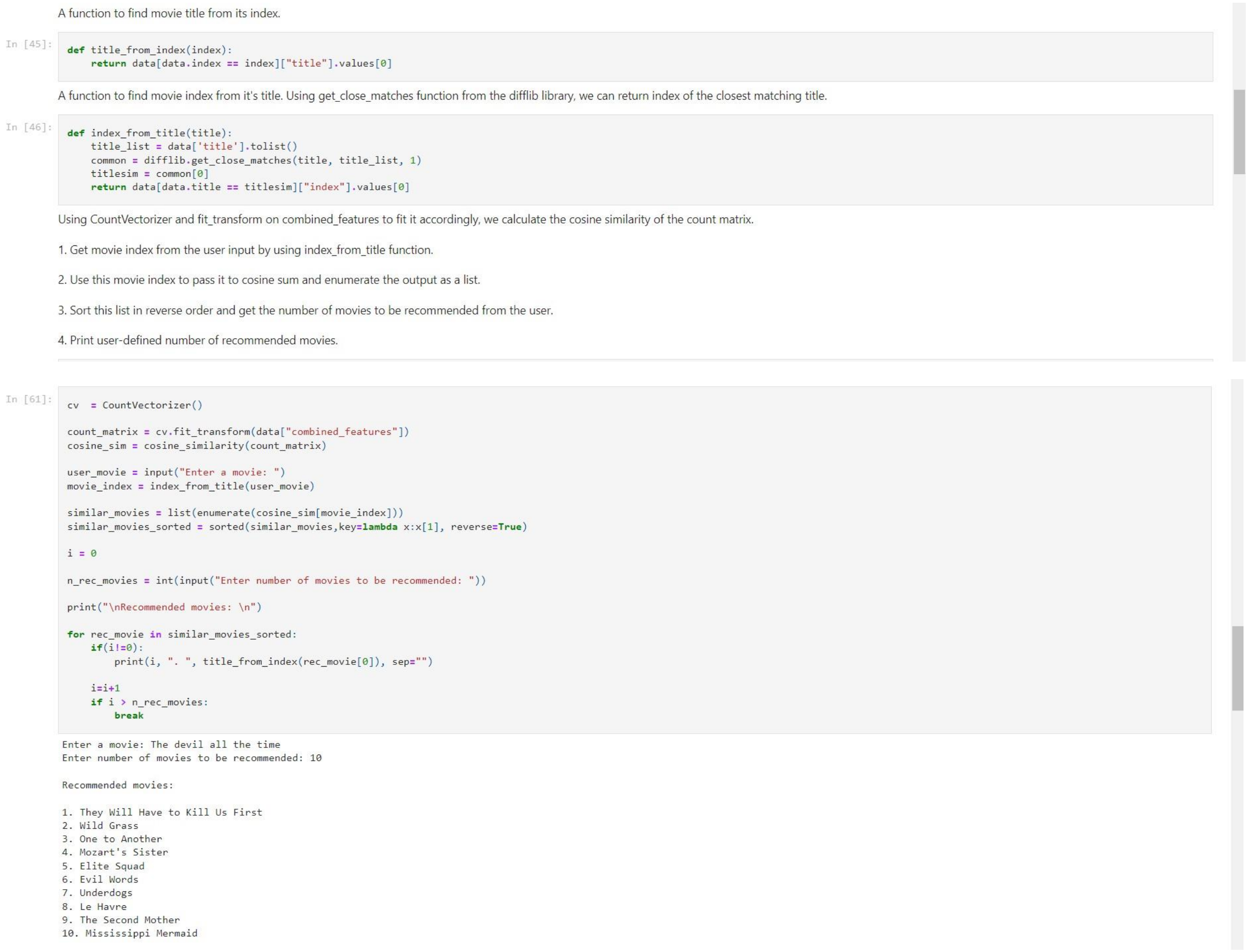
In [44]:

```python
features = ['keywords', 'cast', 'genres', 'director', 'tagline']

for feature in features:
    data[feature] = data[feature].fillna('')

def combine_features(row):
    try:
        return row['keywords'] + " " + row['cast'] + " " + row['genres'] + " " + row['director'] + " " + row['tagline']
    except:
        print("Error:", row)

data["combined_features"] = data.apply(combine_features, axis=1)
```



A function to find movie title from its index.

```
In [45]:
def title_from_index(index):
    return data[data.index == index]["title"].values[0]
```

A function to find movie index from it's title. Using get_close_matches function from the difflib library, we can return index of the closest matching title.

```
In [46]:
def index_from_title(title):
    title_list = data['title'].tolist()
    common = difflib.get_close_matches(title, title_list, 1)
    titlesim = common[0]
    return data[data.title == titlesim]["index"].values[0]
```

Using CountVectorizer and fit_transform on combined_features to fit it accordingly, we calculate the cosine similarity of the count matrix.

1. Get movie index from the user input by using index_from_title function.

2. Use this movie index to pass it to cosine sum and enumerate the output as a list.

3. Sort this list in reverse order and get the number of movies to be recommended from the user.

4. Print user-defined number of recommended movies.

```
In [61]:
cv  = CountVectorizer()

count_matrix = cv.fit_transform(data["combined_features"])
cosine_sim = cosine_similarity(count_matrix)

user_movie = input("Enter a movie: ")
movie_index = index_from_title(user_movie)

similar_movies = list(enumerate(cosine_sim[movie_index]))
similar_movies_sorted = sorted(similar_movies,key=lambda x:x[1], reverse=True)

i = 0

n_rec_movies = int(input("Enter number of movies to be recommended: "))

print("\nRecommended movies: \n")

for rec_movie in similar_movies_sorted:
    if(i!=0):
        print(i, ". ", title_from_index(rec_movie[0]), sep="")

    i=i+1
    if i > n_rec_movies:
        break
```

```
Enter a movie: The devil all the time
Enter number of movies to be recommended: 10

Recommended movies:

1. They Will Have to Kill Us First
2. Wild Grass
3. One to Another
4. Mozart's Sister
5. Elite Squad
6. Evil Words
7. Underdogs
8. Le Havre
9. The Second Mother
10. Mississippi Mermaid
```

Conclusion:

Hence we successfully implemented a movie recommendation system using scikit-learn library in Python.