# LP3 Machine Learning Mini Project

Problem Statemet - To prepare a movie recommendation system which would recommend similar movies to the user according to the user interest. Dataset used - 'movies.csv'

# Group Members -

1. Sakshi Pandit – B-45

2. Pushkaraj Thorat – B-46

3. Vedanti Sarwade – B-47

4. Vaishnavi Kapse – B-48

import numpy as np import pandas as pd import·difflib

from·sklearn.feature\_extraction.text·import·TfidfVectorizer from sklearn.metrics.pairwise import cosine\_similarity

# Data Collection and Preprocessing

movies\_data = pd.read\_csv('movies.csv')

# printing the first 5 rows of the dataframe movies\_data.head()

**index budget genres homepage id keywor**

**0** 0 237000000

**1** 1 300000000

Action Adventure Fantasy

Science Fiction

Adventure Fantasy Action

<http://www.avatarmovie.com/> 19995

<http://disney.go.com/disneypictures/pirates/> 285

cultu cla futu spa

w spa colo

so

oce dr abu exo isla

east ind

trad

Action

**2** 2 245000000 Adventure <http://www.sonypictures.com/movies/spectre/> 206647 secr Crime age

sequ

m

# number of rows and columns in the data frame movies\_data.shape

s based

nov

(4803, 24)

Action comi

m

cri

**3** 3 250000000 Crime <http://www.thedarkknightrises.com/> 49026 fight Drama terror

Thriller secr

ident

based

nov

# selecting the relevant features for recommendation

selected\_features = ['genres','keywords','tagline','cast','director'] print(selected\_features)

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

Action

ma

**4** 4 260000000 Adventure <http://movies.disney.com/john-carter> 49529 medalli Science spa

Fiction trav

pri

# replacing the null valuess with null string

for feature in selected\_features:

movies\_data[feature] = movies\_data[feature].fillna('')

5 rows × 24 columns

# combining all the 5 selected features

combined\_features = movies\_data['genres']+' '+movies\_data['keywords']+' '+movies\_data['tag

print(combined\_features)

1. Action Adventure Fantasy Science Fiction cultu...
2. Adventure Fantasy Action ocean drug abuse exot...
3. Action Adventure Crime spy based on novel secr...
4. Action Crime Drama Thriller dc comics crime fi...
5. 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

# converting the text data to feature vectors vectorizer = TfidfVectorizer()

feature\_vectors = vectorizer.fit\_transform(combined\_features)

print(feature\_vectors)

|  |  |  |  |
| --- | --- | --- | --- |
| (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 |

# Cosine Similarity

# getting the similarity scores using cosine similarity similarity = cosine\_similarity(feature\_vectors)

print(similarity)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| [[1. | 0.07219487 | 0.037733 ... 0. | 0. | 0. | ] |
| [0.07219487 | 1. | 0.03281499 ... 0.03575545 | 0. | 0. | ] |
| [0.037733 | 0.03281499 | 1. ... 0. | 0.05389661 | 0. | ] |
| ... |  |  |  |  |  |
| [0. | 0.03575545 | 0. ... 1. | 0. | 0.02651502] | |
| [0. | 0. | 0.05389661 ... 0. | 1. | 0. ] | |
| [0. | 0. | 0. ... 0.02651502 | 0. | 1. ]] | |

print(similarity.shape) (4803, 4803)

# Getting Movie name

# getting the movie name from the user movie\_name = 'iron man'

# creating a list with all the movie names given in the dataset

list\_of\_all\_titles = movies\_data['title'].tolist() print(list\_of\_all\_titles)

['Avatar', "Pirates of the Caribbean: At World's End", 'Spectre', 'The Dark Knight R

# finding the close match for the movie name given by the user

find\_close\_match = difflib.get\_close\_matches(movie\_name, list\_of\_all\_titles) print(find\_close\_match)

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

close\_match = find\_close\_match[0]

print(close\_match)

Iron Man

# finding the index of the movie with title

index\_of\_the\_movie = movies\_data[movies\_data.title == close\_match]['index'].values[0] print(index\_of\_the\_movie)

68

# getting a list of similar movies

similarity\_score = list(enumerate(similarity[index\_of\_the\_movie])) print(similarity\_score)

[(0, 0.033570748780675445), (1, 0.0546448279236134), (2, 0.013735500604224323), (3,

len(similarity\_score) 4803

# sorting the movies based on their similarity score

sorted\_similar\_movies = sorted(similarity\_score, key = lambda x:x[1], reverse = True) print(sorted\_similar\_movies)

[(68, 1.0000000000000002), (79, 0.40890433998005965), (31, 0.31467052449477506), (7,

# print the name of similar movies based on the index print('Movies suggested for you : \n')

i = 1

for movie in sorted\_similar\_movies: index = movie[0]

title\_from\_index = movies\_data[movies\_data.index==index]['title'].values[0] if (i<30):

print(i, '.',title\_from\_index) i+=1

Movies 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
12. . The Incredible Hulk
13. . The Helix... Loaded
14. . X-Men: First Class
15. . X-Men: Days of Future Past
16. . Captain America: The First Avenger
17. . Kick-Ass 2
18. . Guardians of the Galaxy
19. . Deadpool
20. . Thor: The Dark World
21. . G-Force
22. . X-Men: The Last Stand
23. . Duets
24. . Mortdecai
25. . The Last Airbender
26. . Southland Tales
27. . Zathura: A Space Adventure
28. . Sky Captain and the World of Tomorrow

# Movie Recommendation System

movie\_name = input(' Enter your favourite movie name : ') list\_of\_all\_titles = movies\_data['title'].tolist()

find\_close\_match = difflib.get\_close\_matches(movie\_name, list\_of\_all\_titles) close\_match = find\_close\_match[0]

index\_of\_the\_movie = movies\_data[movies\_data.title == close\_match]['index'].values[0] similarity\_score = list(enumerate(similarity[index\_of\_the\_movie]))

sorted\_similar\_movies = sorted(similarity\_score, key = lambda x:x[1], reverse = True) print('Movies suggested for you : \n')

i = 1

for movie in sorted\_similar\_movies: index = movie[0]

title\_from\_index = movies\_data[movies\_data.index==index]['title'].values[0] if (i<30):

print(i, '.',title\_from\_index) i+=1

Enter your favourite movie name : iron man Movies 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
12. . The Incredible Hulk
13. . The Helix... Loaded
14. . X-Men: First Class
15. . X-Men: Days of Future Past
16. . Captain America: The First Avenger
17. . Kick-Ass 2
18. . Guardians of the Galaxy
19. . Deadpool
20. . Thor: The Dark World
21. . G-Force
22. . X-Men: The Last Stand
23. . Duets
24. . Mortdecai
25. . The Last Airbender
26. . Southland Tales
27. . Zathura: A Space Adventure
28. . Sky Captain and the World of Tomorrow

[Colab paid products](https://colab.research.google.com/signup?utm_source=footer&utm_medium=link&utm_campaign=footer_links) - [Cancel contracts here](https://colab.research.google.com/cancel-subscription) 0s completed at 9:33 PM