GROUP 3

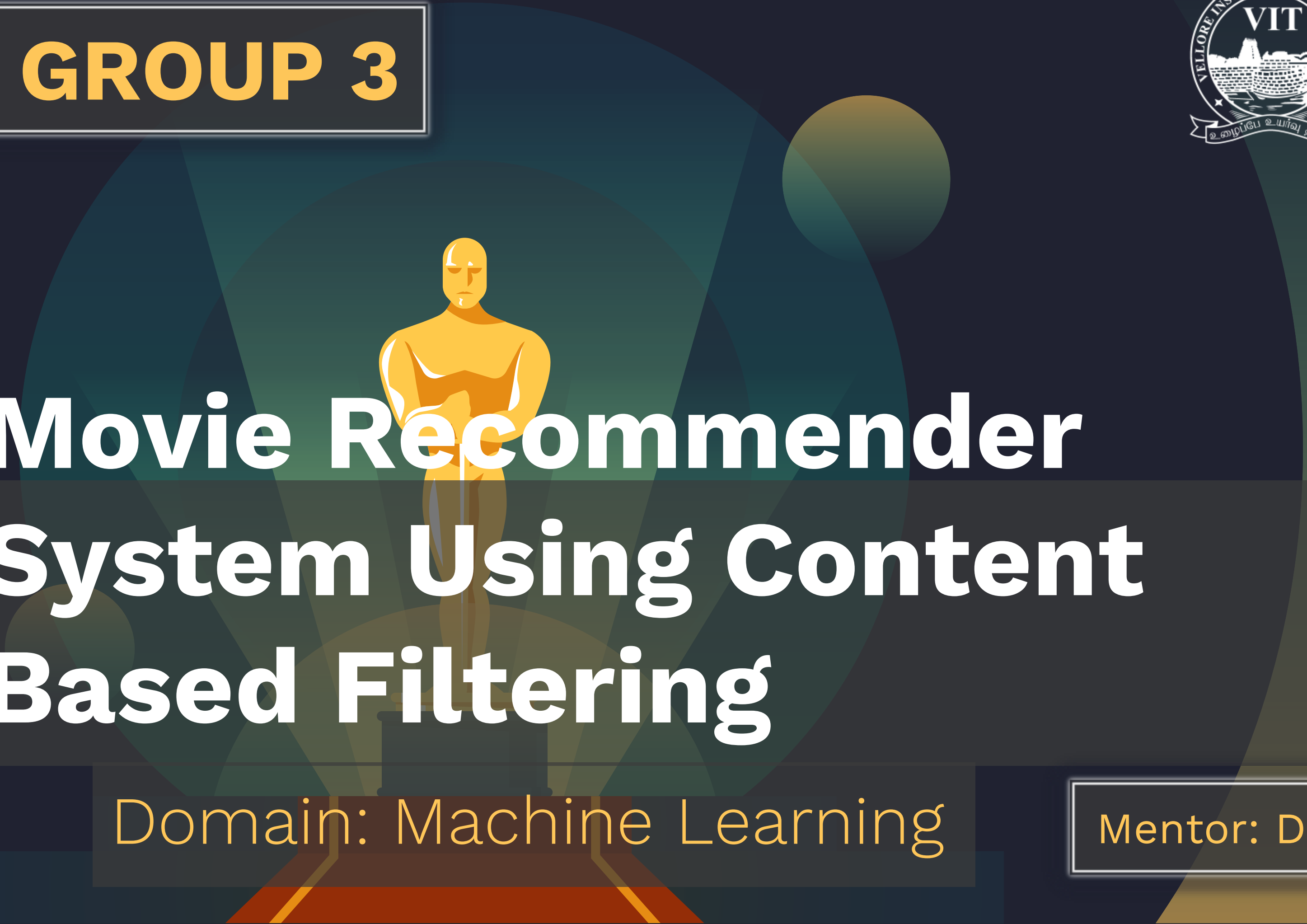
VIT BHOPAL

www.vitbhopal.ac.in

Movie Recommender System Using Content Based Filtering

Domain: Machine Learning

Mentor: Dr. G Elangovan



Team Members:

Aryan Kashyap
20BAI10072

Shrey Khanduja
20BAI10194

Sagar Maheshwari
20BAI10339

Alok Khansali
20BAI10347

Introduction

Movie Recommender System is a system where anyone can write the name of the movie and related to the name user is going to get the similar movies.

The good thing about the project is that it is a reusable project, if in future someone wants to recommend some different commodity then the person just need to change the dataset and the it is going to work as same as it was working before.

Content-Based Movie Recommendation Systems

Content-Based recommender system tries to guess the features or behavior of a user given the item's features, he/she reacts positively to.
Content-based methods are based on the similarity of movie attributes. Using this type of recommender system, if a user watches one movie, similar movies are recommended. For example, if a user watches a comedy movie starring Adam Sandler, the system will recommend them movies in the same genre or starring the same actor, or both. With this in mind, the input for building a content-based recommender system is movie attributes.

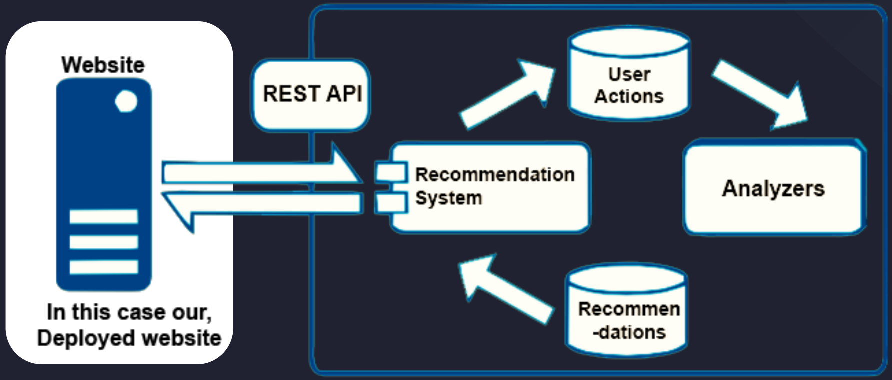
d by author)

- Once, we know the likings of the user we can embed him/her in an embedding space using the feature vector generated and recommend him/her according to his/her choice. During recommendation, the similarity metrics (We will talk about it in a bit) are calculated from the item’s feature vectors and the user’s preferred feature vectors from his/her previous records. Then, the top few are recommended.

- Content-based filtering does not require other user’s data during recommendations to one user.

How do Content Based Recommender Systems work?

- A content based recommender works with data that the user provides, either explicitly (rating) or implicitly (clicking on a link). Based on that data, a user profile is generated, which is then used to make suggestions to the user. As the user provides more inputs or takes actions on the recommendations, the engine becomes more and more accurate.



The Dataset

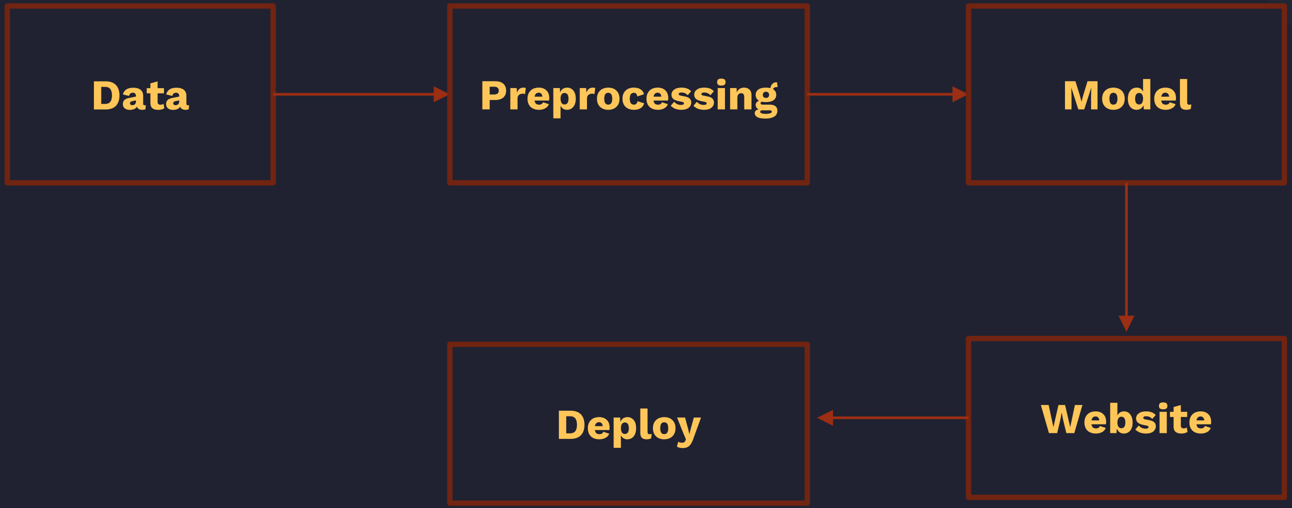
For the system, we have used the open-source TMDB Movie dataset from Kaggle. This dataset contains 5000 movies of various cast and crew.

We will create three columns from the data

After preprocessing of initial dataset:

- title
- movieId
- tags

Overall Flow Diagram



Overall System Architecture:

A common architecture of Recommender Systems comprises of the following three essential components:

1. Candidate Generation

This is the first stage of the Recommender Systems and takes events from the user’s past activity as input and retrieves a small subset (hundreds) of videos from a large corpus. There are mainly two common candidate generation approaches:

- Content-Based Filtering
- Collaborative Filtering

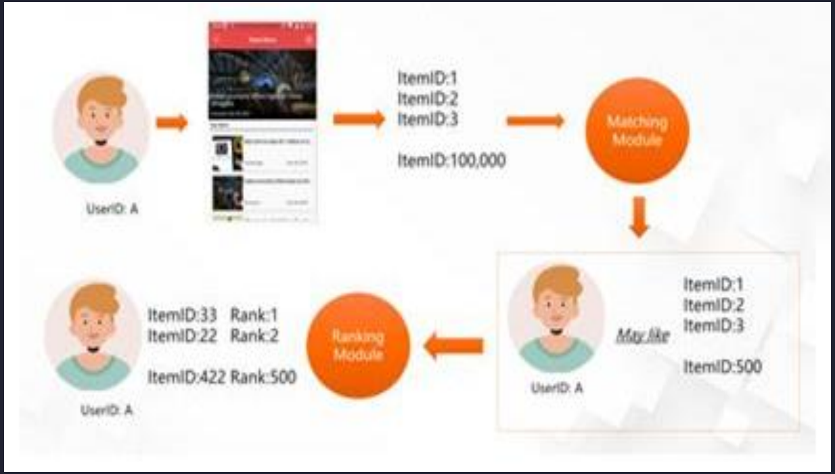
2. Scoring

This constitutes the second stage where another model further ranks and scores the candidates usually on a scale of 10.

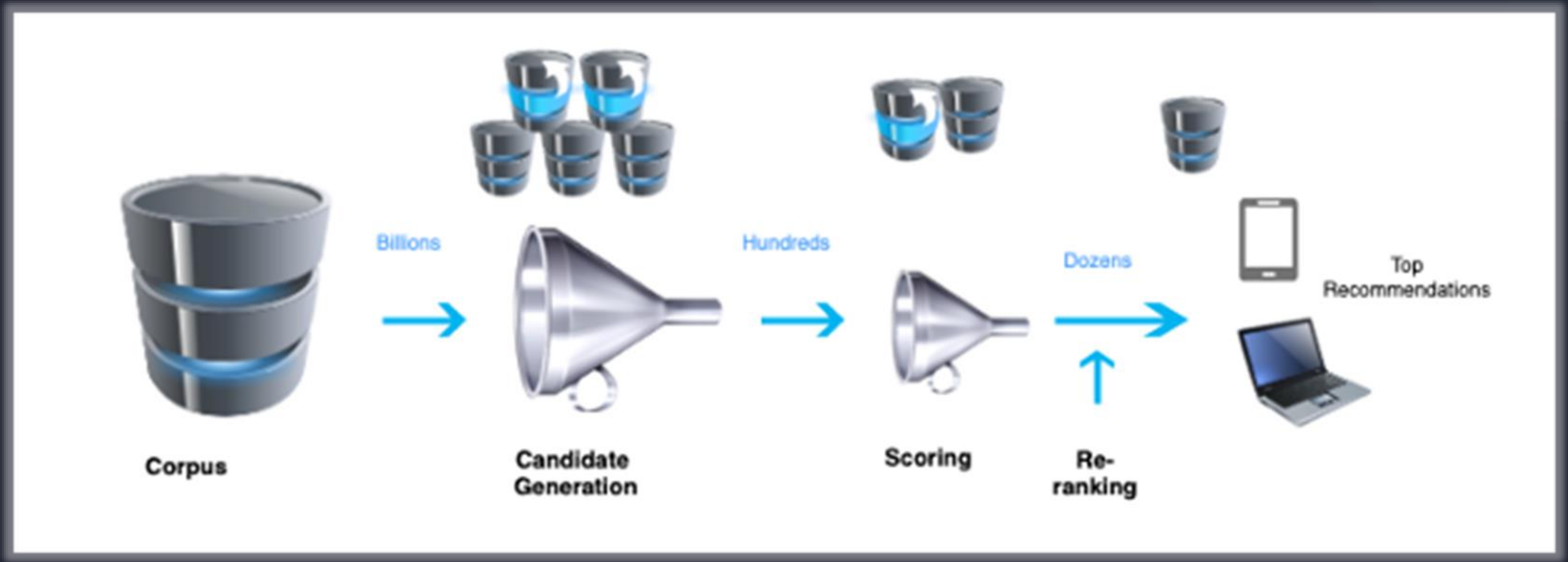
For instance, in the case of YouTube, the ranking network accomplishes this task by assigning a score to each video according to the desired objective function using a rich set of features describing the video and user. The highest scoring videos are presented to the user, ranked by their score.

3. Re-ranking

In the third stage, the system takes into account additional constraints to ensure diversity, freshness, and fairness. For instance, the system removes the content which has been explicitly disliked by the user earlier and also takes into account any fresh item on the site.



Architecture Diagram:



Novelty of the project

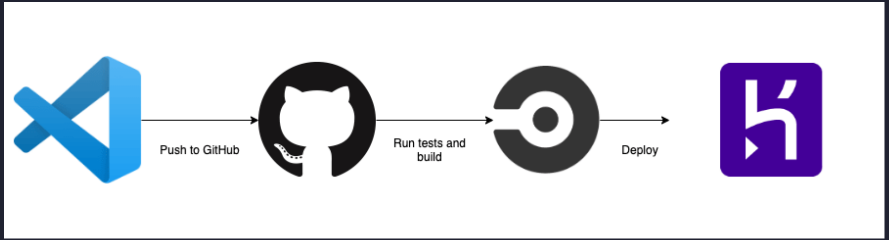
“ What makes the project unique? ”

Novelty of the project

Content-based movie recommendation system recommends movies similar to the movie user likes and analyses the sentiments on the reviews given by the user for that movie.

Distinct features:

- Content based recommendation system
- Reusable with any datasets
- Deployment on Heroku.
- Ready to use website



Real time usage

“ Application of this project?
”

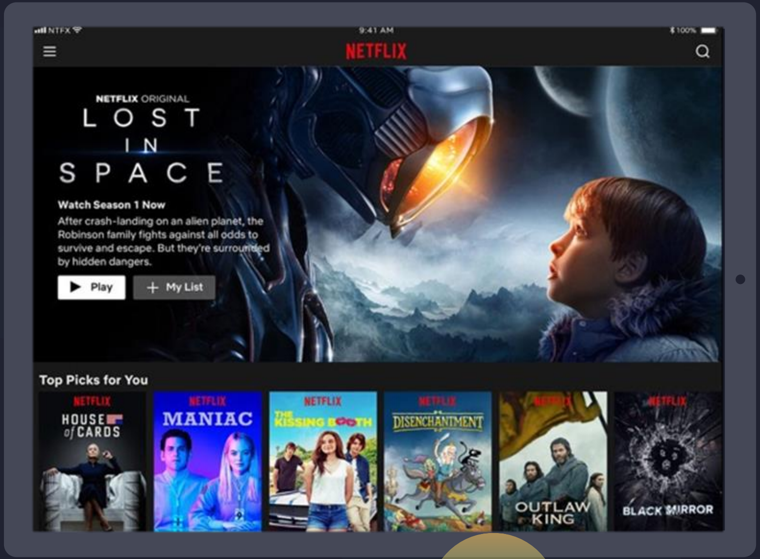
Real time usage:

The rapid growth of data collection has led to a new era of information.

We now live in what some call the “era of abundance”. For any given product, there are sometimes thousands of options to choose from. Think of the examples above: streaming videos, social networking, online shopping; the list goes on.

Recommendation Systems are a type of information filtering systems as they improve the quality of search results and provides items that are more relevant to the search item or are related to the search history of the user.

Almost every major tech company has applied them in some form or the other. From a business standpoint, the more relevant products a user finds on the platform, the higher their engagement. This often results in increased revenue for the platform itself. Various sources say that as much as 35–40% of tech giants’ revenue comes from recommendations alone.



Tablet App

- Amazon uses it to suggest products to customers, YouTube uses it to decide which video to play next on autoplay, and Facebook uses it to recommend pages to like and people to follow.

- Companies like Netflix and Spotify depend highly on the effectiveness of their recommendation engines for their business and success.

Hardware & Software requirements

“ The Project would run on every machine? ”

Hardware & Software requirements:

Hardware:

1. CPU: 2 x 64-bit 2.8 GHz
2. Ram: Systems with 2GB RAM (4GB preferable)
3. Storage: 10gb (Preferred)

Software:

1. Operating system: Linux- Ubuntu 16.04 to 17.10, Windows 8 to 11 or MacOS
2. Browser: Google Chrome or Mozilla Firefox (latest version)
3. Anaconda Navigator with Python 3.6 (Latest Preferred)
4. Python Modules such as NumPy, Pandas, Scikit-learn & Nltk

Module Description and Workflow

Bag of Words (BOW) Model

The bag-of-words (BOW) model is a representation that turns arbitrary text into fixed-length vectors by counting how many times each word appears. This process is often referred to as vectorization.

Let’s understand this with an example. Suppose we wanted to vectorize the following:

- ➔ the cat sat
- ➔ the cat sat in the hat
- ➔ the cat with the hat

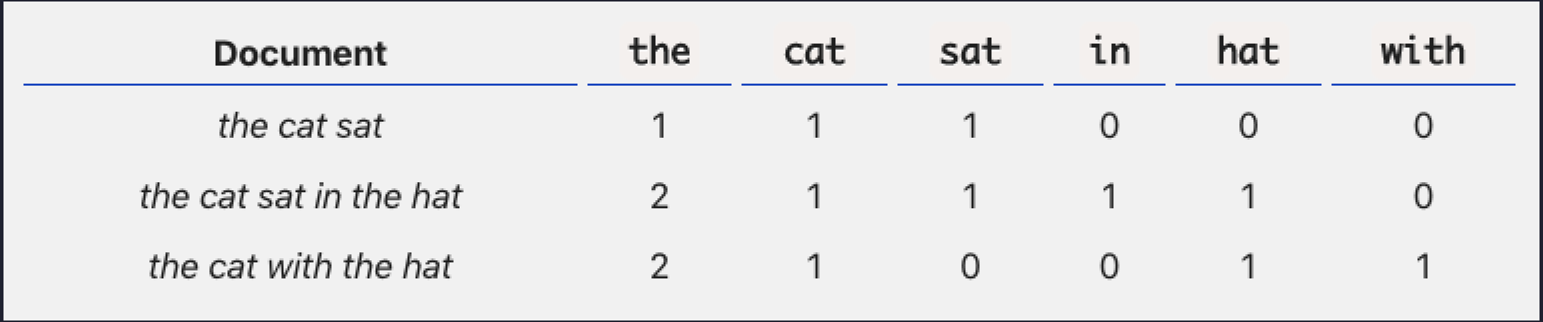
We’ll refer to each of these as a text document.

Step 1: Determine the Vocabulary [tokenization]

We first define our vocabulary, which is the set of all words found in our document set. The only words that are found in the 3 documents above are: **the**, **cat**, **sat**, **in**, **the**, **hat**, and **with**.

Step 2: Count

To vectorize our documents, all we have to do is count how many times each word appears:



Document	the	cat	sat	in	hat	with
the cat sat	1	1	1	0	0	0
the cat sat in the hat	2	1	1	1	1	0
the cat with the hat	2	1	0	0	1	1

1. NumPy

Numpy is a general-purpose array-processing package that provides a high-performance multidimensional array object, and tools for working with these arrays. It is one of the most fundamental package for scientific computing with Python.

Functions used:

- ***ndim(): return the number of dimensions of an array.***
- ***shape(): returns a tuple with each index having the number of corresponding elements.***

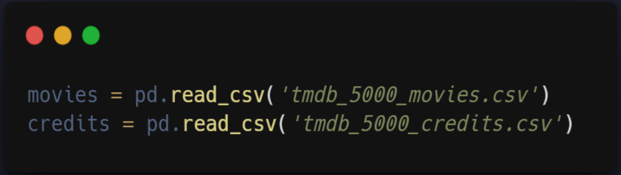
2. Pandas

Pandas is a Python package that offers various data structures and operations for manipulating numerical data and time series.

It is mainly popular for importing and analyzing data much easier.

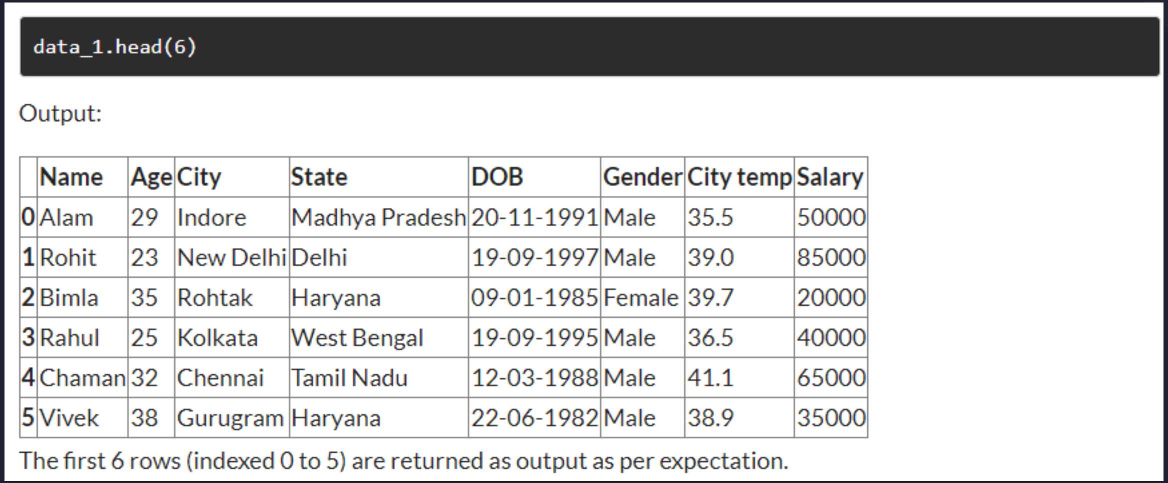
Functions used:

- ***read_csv(): Loads the CSV into a DataFrame***



```
movies = pd.read_csv( 'tmdb_5000_movies.csv')
credits = pd.read_csv( 'tmdb_5000_credits.csv')
```

- ***head(): It returns top n (5 by default) rows of a data frame.***



```
data_1.head(6)
```

Output:

	Name	Age	City	State	DOB	Gender	City temp	Salary
0	Alam	29	Indore	Madhya Pradesh	20-11-1991	Male	35.5	50000
1	Rohit	23	New Delhi	Delhi	19-09-1997	Male	39.0	85000
2	Bimla	35	Rohtak	Haryana	09-01-1985	Female	39.7	20000
3	Rahul	25	Kolkata	West Bengal	19-09-1995	Male	36.5	40000
4	Chaman	32	Chennai	Tamil Nadu	12-03-1988	Male	41.1	65000
5	Vivek	38	Gurugram	Haryana	22-06-1982	Male	38.9	35000

The first 6 rows (indexed 0 to 5) are returned as output as per expectation.

- ***dropna(): This method allows the user to analyze and drop Rows/Columns with Null values in different ways.***

3. OS

OS module in Python provides functions for interacting with the operating system. It comes under Python’s standard utility modules and this module provides a portable way of using operating system-dependent functionality which include many functions to interact with the file system.

Functions used:

- ***walk(): It generates the file names in a directory tree by walking the tree either top-down or bottom-up.***
- ***path(): It's another Python module, which also provides a big range of useful methods to manipulate files and directories.***

4. nltk

NLTK stands for Natural Language Toolkit and it is suite of libraries and programs in Python for Natural Language Processing Tasks. It is one of the most widely used NLP Python libraries.

It can perform various NLP tasks like tokenization, stemming, POS tagging, lemmatization and classification to name a few.

Functions used:

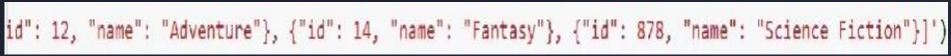
- ***PorterStemmer(): Its an algorithm used for removing the commoner morphological and inflexional endings from words***

5. ast

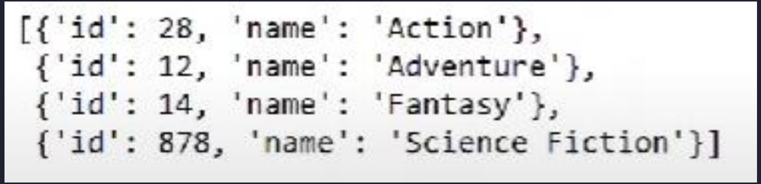
The ***ast*** module helps Python applications to process trees of the Python abstract syntax grammar. The abstract syntax itself might change with each Python release; this module helps to find out programmatically what the current grammar looks like.

ast.literal_eval: Safely evaluate an expression node or a string containing a Python literal or container display.

List of String



```
id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}, {"id": 878, "name": "Science Fiction"}]')
```



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

List

6. Scikit-learn

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python.

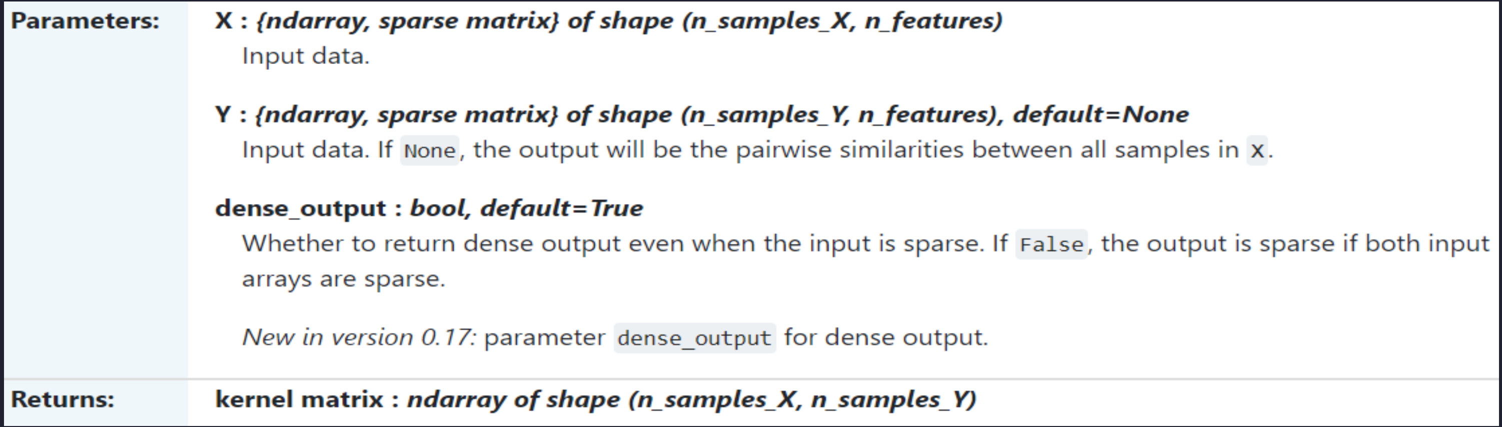
Used classes:

- ***countVectorizer***
- ***cosine_similarity***

- **sklearn.feature_extraction.text.CountVectorizer** : Convert a collection of text documents to a matrix of token counts.
- **sklearn.metrics.pairwise.cosine_similarity** : Compute 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:

$$K(X, Y) = \langle X, Y \rangle / (||X||*||Y||)$$



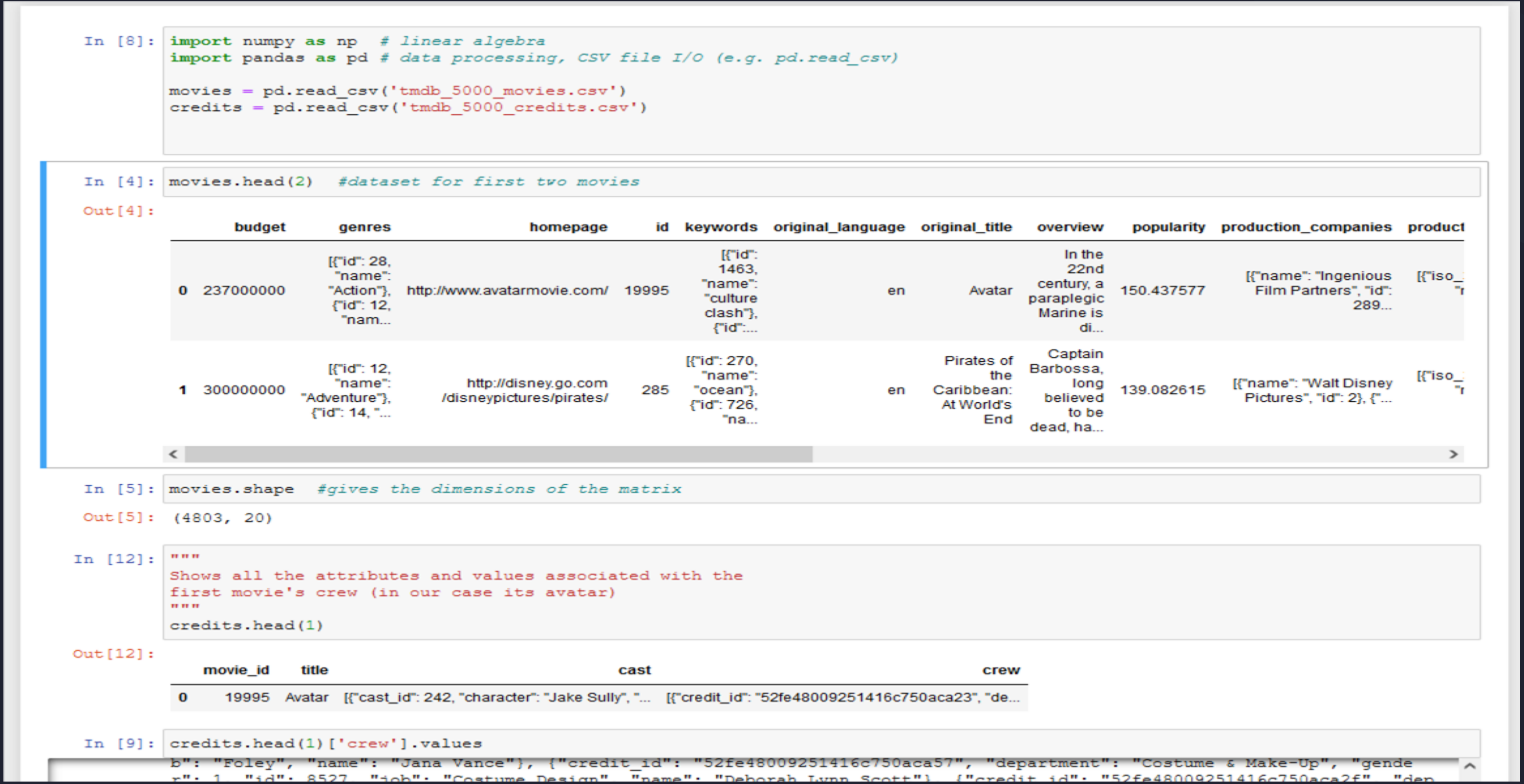
Parameters:	**X : *{ndarray, sparse matrix} of shape (n_samples_X, n_features)*** Input data. **Y : *{ndarray, sparse matrix} of shape (n_samples_Y, n_features), default=None*** Input data. If `None`, the output will be the pairwise similarities between all samples in `X`. **dense_output : *bool, default=True*** Whether to return dense output even when the input is sparse. If `False`, the output is sparse if both input arrays are sparse. *New in version 0.17:* parameter `dense_output` for dense output.
Returns:	**kernel matrix : *ndarray of shape (n_samples_X, n_samples_Y)***

7. pickle

The ***`pickle`*** module implements binary protocols for serializing and de-serializing a Python object structure. *"Pickling"* is the process whereby a Python object hierarchy is converted into a byte stream, and *"unpickling"* is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy.

The ***dump()*** method of the pickle module in Python, **converts a Python object hierarchy into a byte stream**. This process is also called as serialization. The converted byte stream can be written to a buffer or to a disk file.

Implementation of the modules:



```
In [8]: import numpy as np  # linear algebra
        import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

        movies = pd.read_csv('tmdb_5000_movies.csv')
        credits = pd.read_csv('tmdb_5000_credits.csv')
```

```
In [4]: movies.head(2)  #dataset for first two movies
```

Out[4]:

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies	product
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"name": "Ingenious Film Partners", "id": 289...	[{"iso_ "r
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com /disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"name": "Walt Disney Pictures", "id": 2}, {"...	[{"iso_ "r

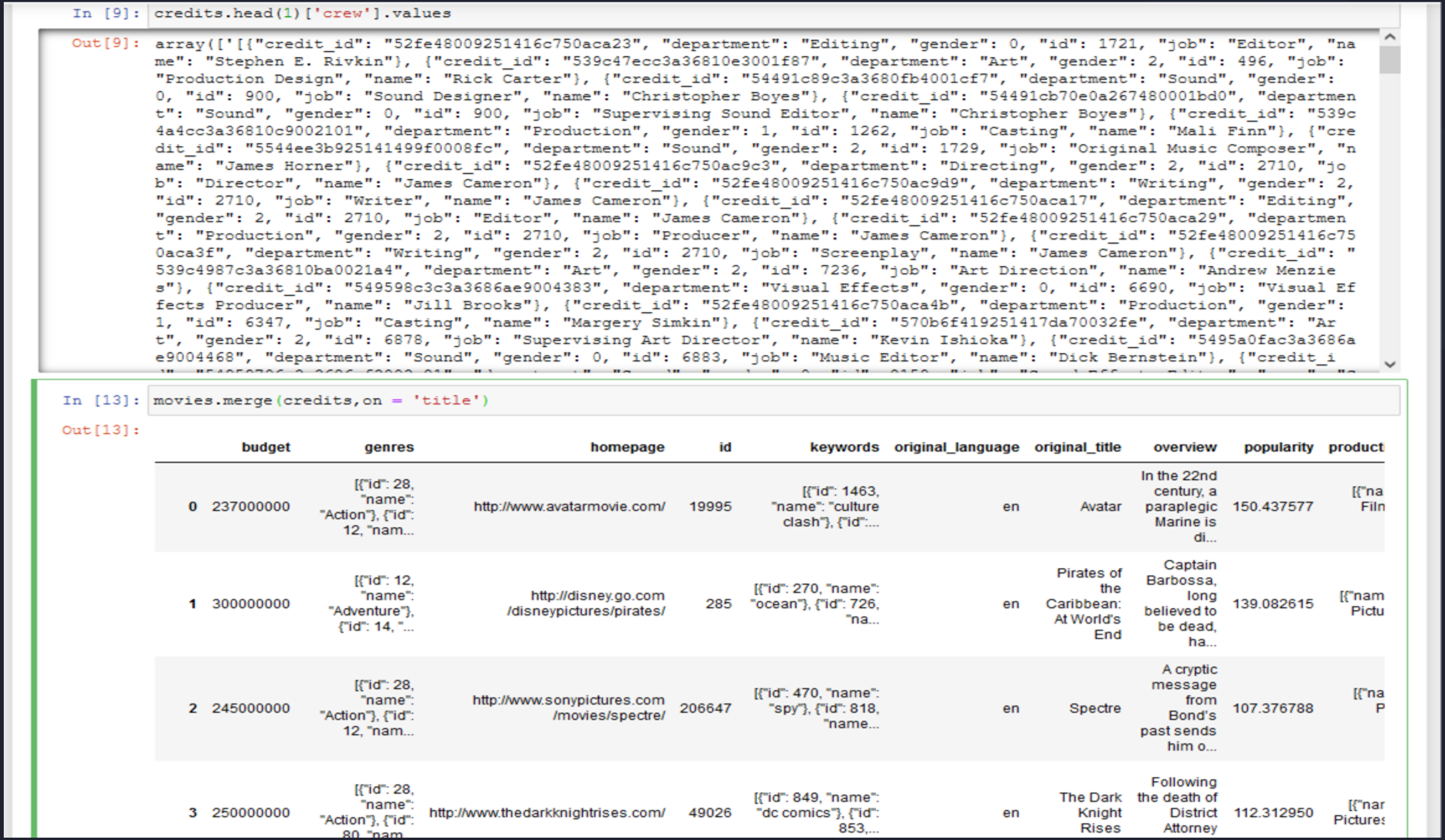
```
In [5]: movies.shape  #gives the dimensions of the matrix
Out[5]: (4803, 20)
```

```
In [12]: """
         Shows all the attributes and values associated with the
         first movie's crew (in our case its avatar)
         """
         credits.head(1)
```

Out[12]:

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...

```
In [9]: credits.head(1)['crew'].values
        b": "Foley", "name": "Jana Vance"}, {"credit_id": "52fe48009251416c750aca57", "department": "Costume & Make-Up", "gende
        r": 1, "id": 8527, "job": "Costume Design", "name": "Deborah Lynn Scott"}, {"credit_id": "52fe48009251416c750aca2f", "dep
```



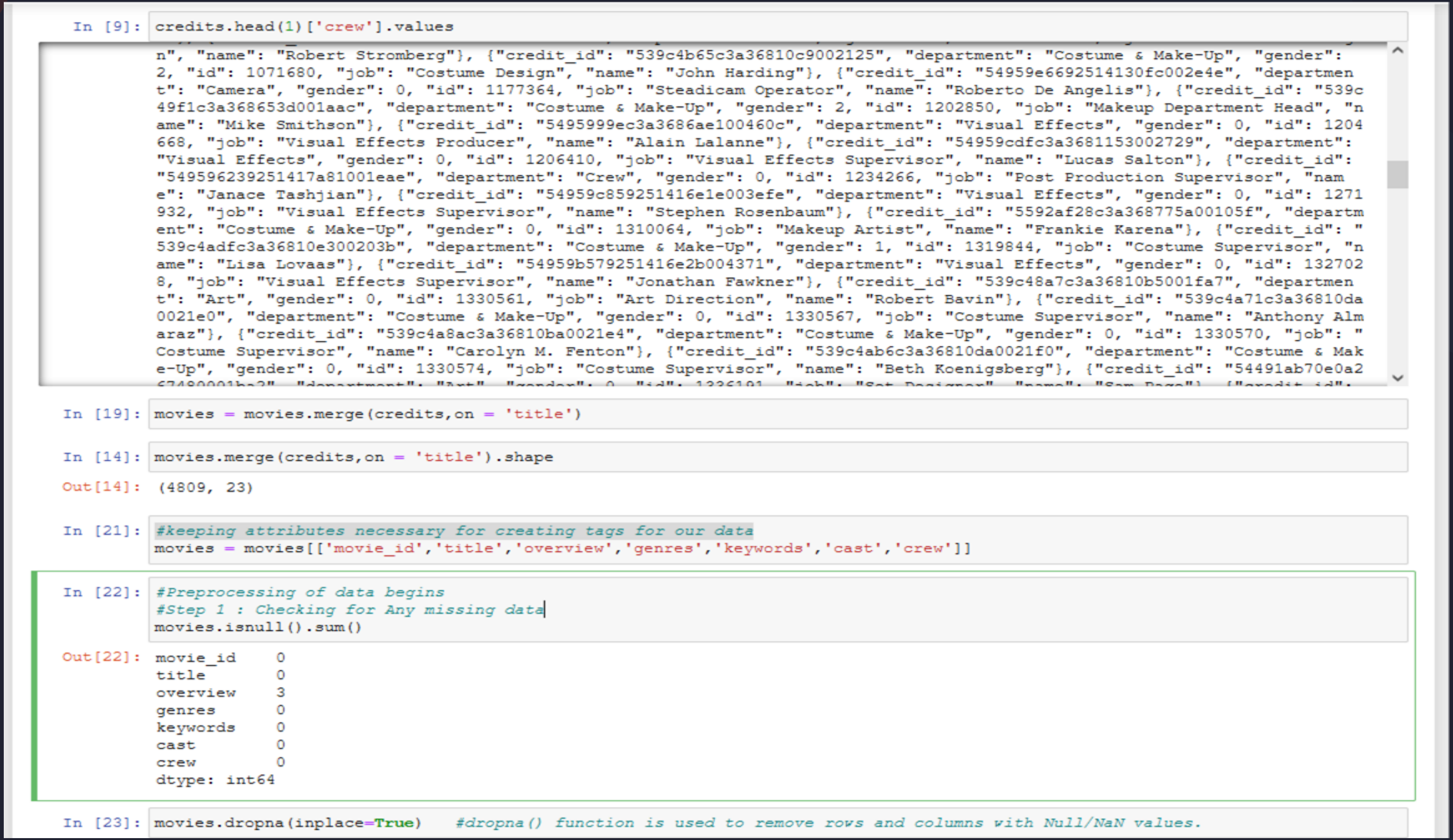
```
In [9]:  credits.head(1)['crew'].values
```

```
Out[9]: array(['[{"credit_id": "52fe48009251416c750aca23", "department": "Editing", "gender": 0, "id": 1721, "job": "Editor", "na
        me": "Stephen E. Rivkin"}, {"credit_id": "539c47ecc3a36810e3001f87", "department": "Art", "gender": 2, "id": 496, "job":
        "Production Design", "name": "Rick Carter"}, {"credit_id": "54491c89c3a3680fb4001cf7", "department": "Sound", "gender":
        0, "id": 900, "job": "Sound Designer", "name": "Christopher Boyes"}, {"credit_id": "54491cb70e0a267480001bd0", "departmen
        t": "Sound", "gender": 0, "id": 900, "job": "Supervising Sound Editor", "name": "Christopher Boyes"}, {"credit_id": "539c
        4a4cc3a36810c9002101", "department": "Production", "gender": 1, "id": 1262, "job": "Casting", "name": "Mali Finn"}, {"cre
        dit_id": "5544ee3b925141499f0008fc", "department": "Sound", "gender": 2, "id": 1729, "job": "Original Music Composer", "n
        ame": "James Horner"}, {"credit_id": "52fe48009251416c750ac9c3", "department": "Directing", "gender": 2, "id": 2710, "jo
        b": "Director", "name": "James Cameron"}, {"credit_id": "52fe48009251416c750ac9d9", "department": "Writing", "gender": 2,
        "id": 2710, "job": "Writer", "name": "James Cameron"}, {"credit_id": "52fe48009251416c750aca17", "department": "Editing",
        "gender": 2, "id": 2710, "job": "Editor", "name": "James Cameron"}, {"credit_id": "52fe48009251416c750aca29", "departmen
        t": "Production", "gender": 2, "id": 2710, "job": "Producer", "name": "James Cameron"}, {"credit_id": "52fe48009251416c75
        0aca3f", "department": "Writing", "gender": 2, "id": 2710, "job": "Screenplay", "name": "James Cameron"}, {"credit_id": "
        539c4987c3a36810ba0021a4", "department": "Art", "gender": 2, "id": 7236, "job": "Art Direction", "name": "Andrew Menzie
        s"}, {"credit_id": "549598c3c3a3686ae9004383", "department": "Visual Effects", "gender": 0, "id": 6690, "job": "Visual Ef
        fects Producer", "name": "Jill Brooks"}, {"credit_id": "52fe48009251416c750aca4b", "department": "Production", "gender":
        1, "id": 6347, "job": "Casting", "name": "Margery Simkin"}, {"credit_id": "570b6f419251417da70032fe", "department": "Ar
        t", "gender": 2, "id": 6878, "job": "Supervising Art Director", "name": "Kevin Ishioka"}, {"credit_id": "5495a0fac3a3686a
        e9004468", "department": "Sound", "gender": 0, "id": 6883, "job": "Music Editor", "name": "Dick Bernstein"}, {"credit_i
```

```
In [13]: movies.merge(credits,on = 'title')
```

Out[13]:

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	product
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"na Filn
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com /disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"nam Pictu
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com /movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"na P
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney	112.312950	[{"nar Pictures



```
In [9]: credits.head(1)['crew'].values
```

```
n", "name": "Robert Stromberg"}, {"credit_id": "539c4b65c3a36810c9002125", "department": "Costume & Make-Up", "gender":
2, "id": 1071680, "job": "Costume Design", "name": "John Harding"}, {"credit_id": "54959e6692514130fc002e4e", "departmen
t": "Camera", "gender": 0, "id": 1177364, "job": "Steadicam Operator", "name": "Roberto De Angelis"}, {"credit_id": "539c
49f1c3a368653d001aac", "department": "Costume & Make-Up", "gender": 2, "id": 1202850, "job": "Makeup Department Head", "n
ame": "Mike Smithson"}, {"credit_id": "5495999ec3a3686ae100460c", "department": "Visual Effects", "gender": 0, "id": 1204
668, "job": "Visual Effects Producer", "name": "Alain Lalanne"}, {"credit_id": "54959cdfc3a3681153002729", "department":
"Visual Effects", "gender": 0, "id": 1206410, "job": "Visual Effects Supervisor", "name": "Lucas Salton"}, {"credit_id":
"549596239251417a81001eae", "department": "Crew", "gender": 0, "id": 1234266, "job": "Post Production Supervisor", "nam
e": "Janace Tashjian"}, {"credit_id": "54959c859251416e1e003efe", "department": "Visual Effects", "gender": 0, "id": 1271
932, "job": "Visual Effects Supervisor", "name": "Stephen Rosenbaum"}, {"credit_id": "5592af28c3a368775a00105f", "departm
ent": "Costume & Make-Up", "gender": 0, "id": 1310064, "job": "Makeup Artist", "name": "Frankie Karena"}, {"credit_id": "
539c4adfc3a36810e300203b", "department": "Costume & Make-Up", "gender": 1, "id": 1319844, "job": "Costume Supervisor", "n
ame": "Lisa Lovaas"}, {"credit_id": "54959b579251416e2b004371", "department": "Visual Effects", "gender": 0, "id": 132702
8, "job": "Visual Effects Supervisor", "name": "Jonathan Fawkner"}, {"credit_id": "539c48a7c3a36810b5001fa7", "departmen
t": "Art", "gender": 0, "id": 1330561, "job": "Art Direction", "name": "Robert Bavin"}, {"credit_id": "539c4a71c3a36810da
0021e0", "department": "Costume & Make-Up", "gender": 0, "id": 1330567, "job": "Costume Supervisor", "name": "Anthony Alm
araz"}, {"credit_id": "539c4a8ac3a36810ba0021e4", "department": "Costume & Make-Up", "gender": 0, "id": 1330570, "job": "
Costume Supervisor", "name": "Carolyn M. Fenton"}, {"credit_id": "539c4ab6c3a36810da0021f0", "department": "Costume & Mak
e-Up", "gender": 0, "id": 1330574, "job": "Costume Supervisor", "name": "Beth Koenigsberg"}, {"credit_id": "54491ab70e0a2
```

```
In [19]: movies = movies.merge(credits,on = 'title')
```

```
In [14]: movies.merge(credits,on = 'title').shape
```

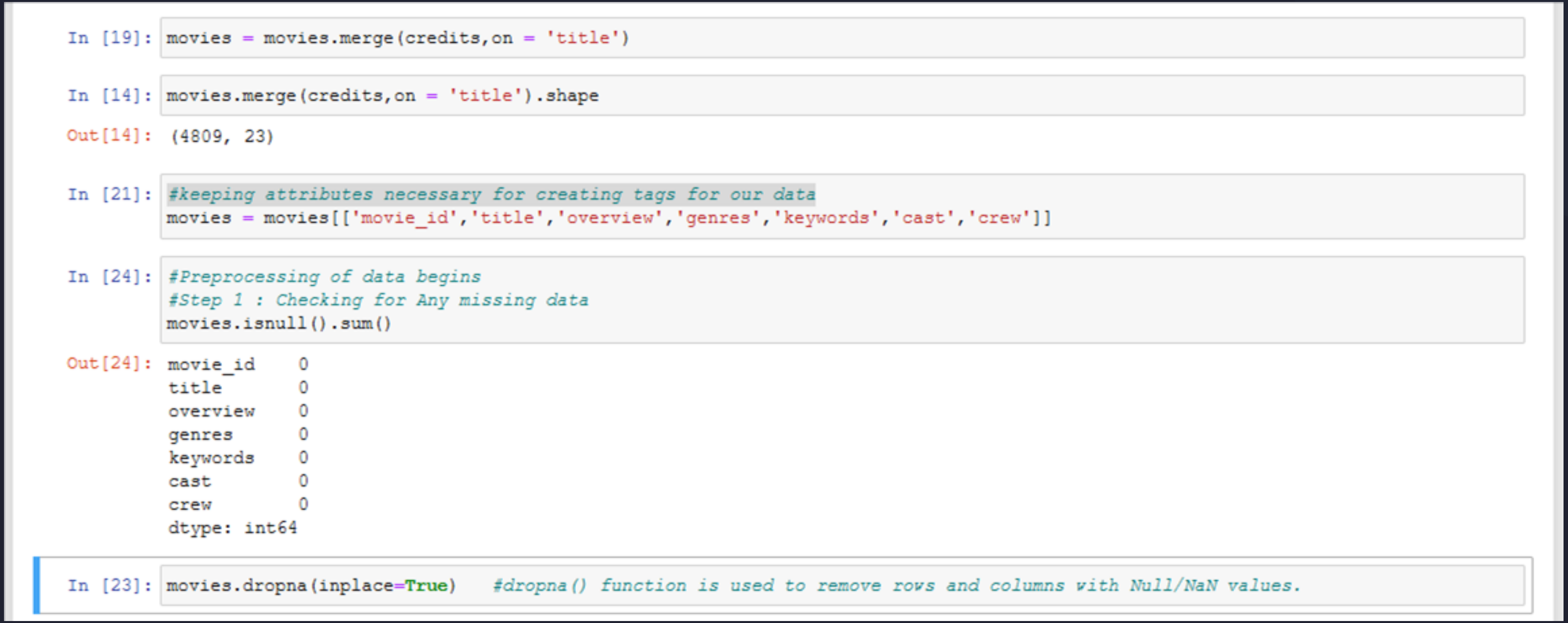
```
Out[14]: (4809, 23)
```

```
In [21]: #keeping attributes necessary for creating tags for our data
         movies = movies[['movie_id','title','overview','genres','keywords','cast','crew']]
```

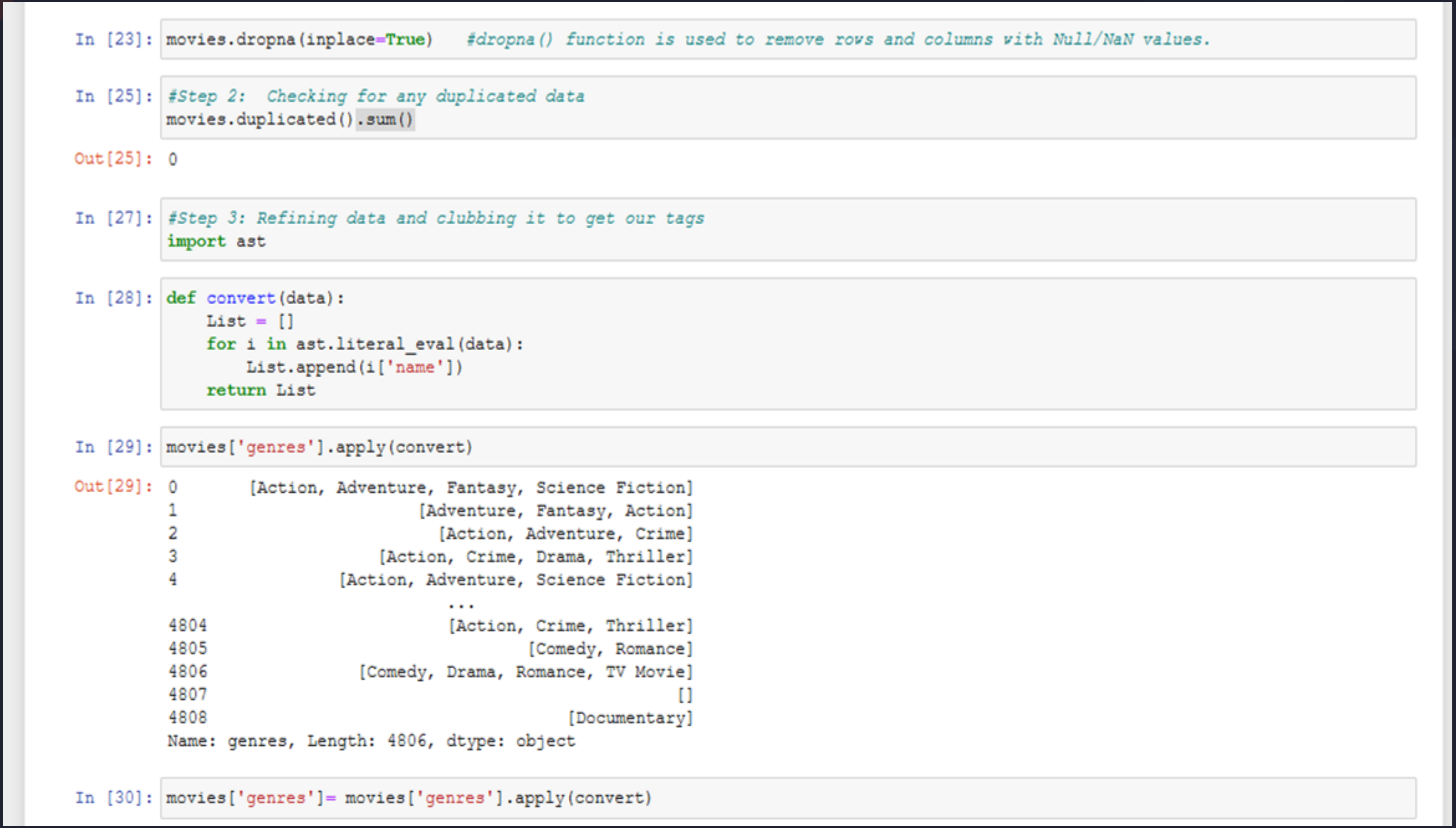
```
In [22]: #Preprocessing of data begins
         #Step 1 : Checking for Any missing data
         movies.isnull().sum()
```

```
Out[22]: movie_id    0
         title       0
         overview    3
         genres      0
         keywords    0
         cast        0
         crew        0
         dtype: int64
```

```
In [23]: movies.dropna(inplace=True)    #dropna() function is used to remove rows and columns with Null/NaN values.
```



```
In [19]: movies = movies.merge(credits,on = 'title')

In [14]: movies.merge(credits,on = 'title').shape

Out[14]: (4809, 23)

In [21]: #keeping attributes necessary for creating tags for our data
         movies = movies[['movie_id','title','overview','genres','keywords','cast','crew']]

In [24]: #Preprocessing of data begins
         #Step 1 : Checking for Any missing data
         movies.isnull().sum()

Out[24]: movie_id    0
         title       0
         overview    0
         genres      0
         keywords    0
         cast        0
         crew        0
         dtype: int64

In [23]: movies.dropna(inplace=True)    #dropna() function is used to remove rows and columns with Null/NaN values.
```



```
In [23]: movies.dropna(inplace=True)    #dropna() function is used to remove rows and columns with Null/NaN values.
```

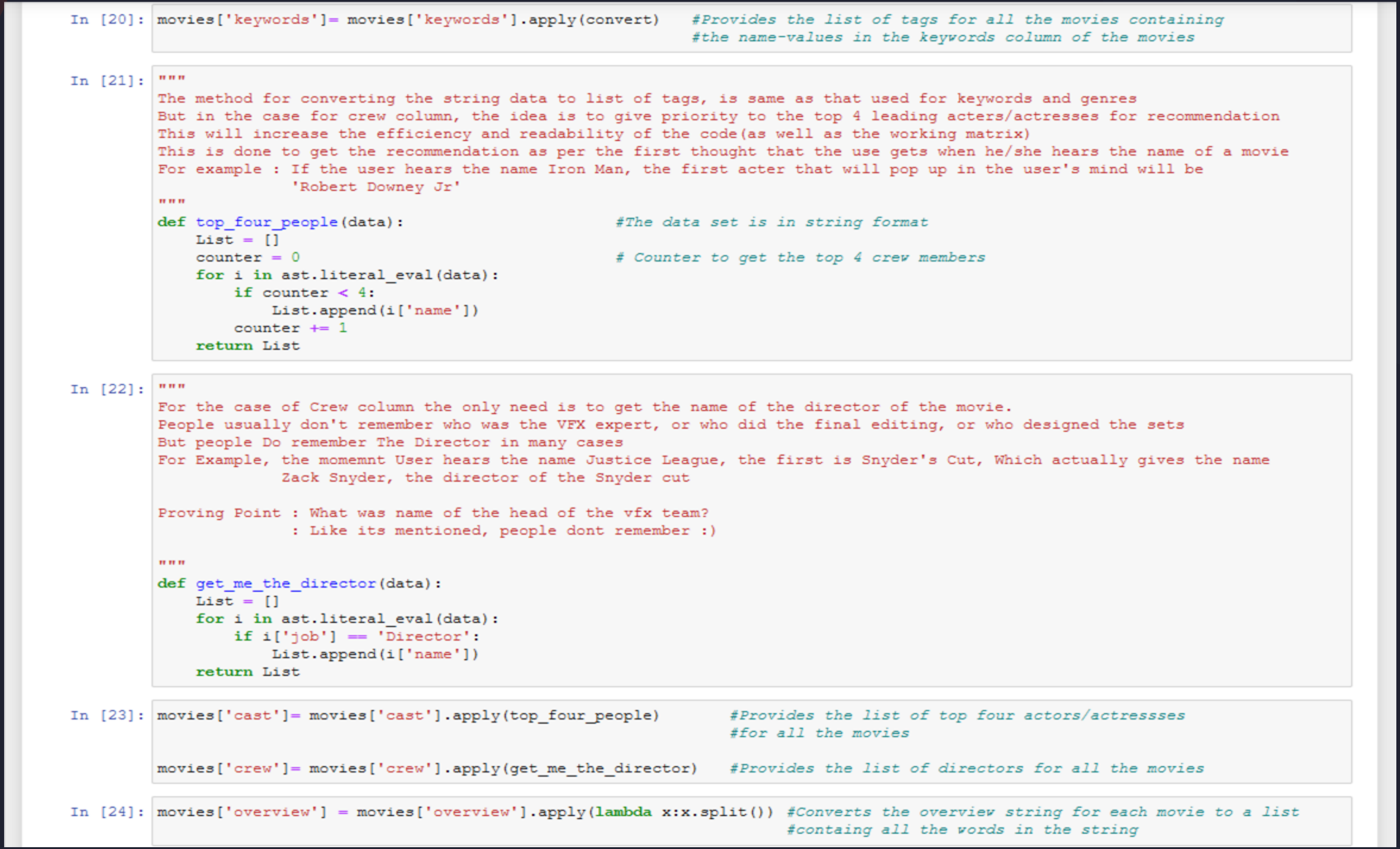
```
In [25]: #Step 2:  Checking for any duplicated data
         movies.duplicated().sum()

Out[25]: 0
```

```
In [27]: #Step 3: Refining data and clubbing it to get our tags
         import ast
```

```
In [28]: def convert(data):
             List = []
             for i in ast.literal_eval(data):
                 List.append(i['name'])
             return List
```

```
In [29]: movies['genres'].apply(convert)

Out[29]: 0        [Action, Adventure, Fantasy, Science Fiction]
         1                       [Adventure, Fantasy, Action]
         2                         [Action, Adventure, Crime]
         3                    [Action, Crime, Drama, Thriller]
         4                [Action, Adventure, Science Fiction]
                                      ...
         4804                        [Action, Crime, Thriller]
         4805                                [Comedy, Romance]
         4806               [Comedy, Drama, Romance, TV Movie]
         4807                                               []
         4808                                    [Documentary]
         Name: genres, Length: 4806, dtype: object
```

```
In [30]: movies['genres']= movies['genres'].apply(convert)
```



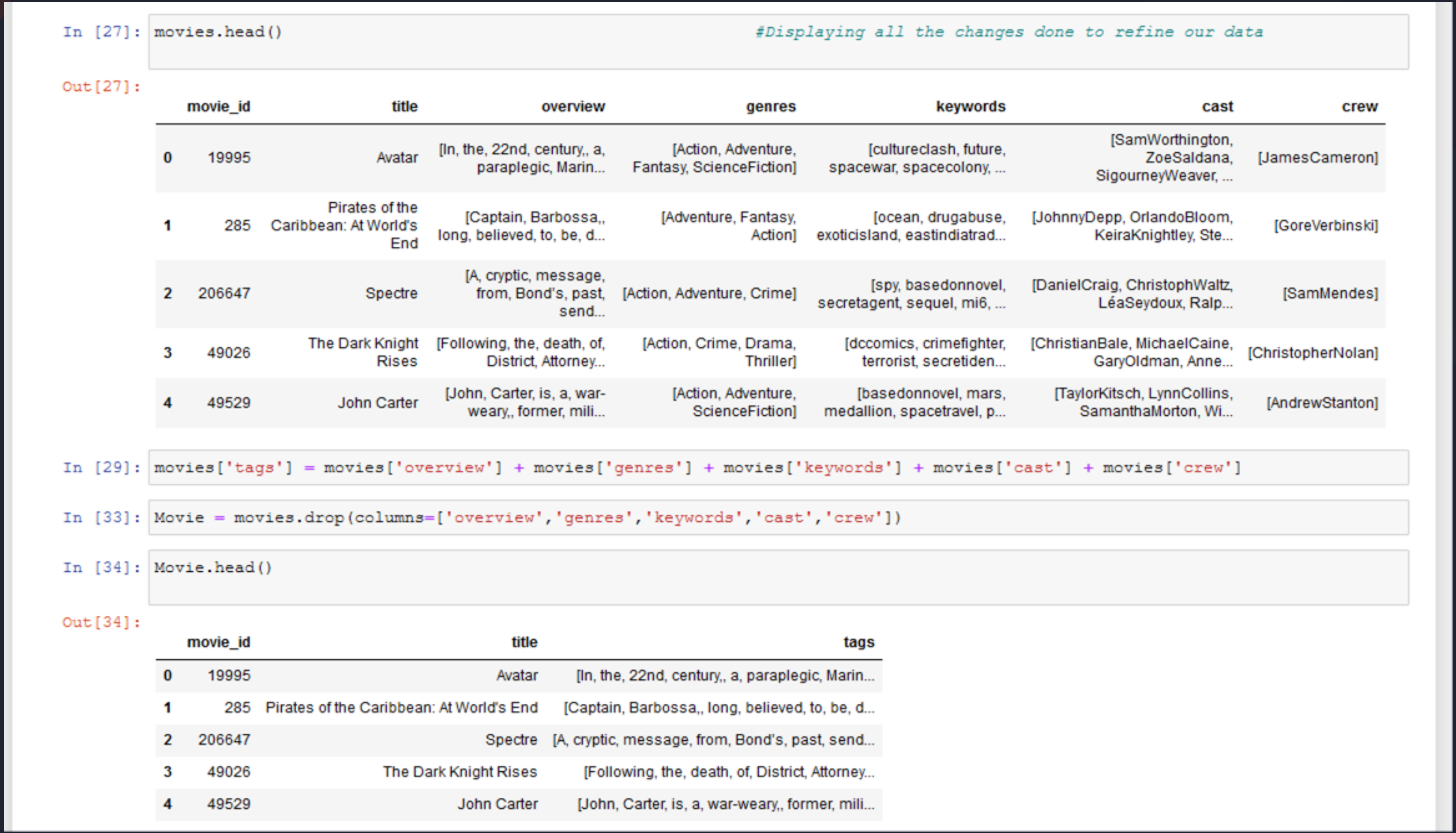
```
In [20]: movies['keywords']= movies['keywords'].apply(convert)    #Provides the list of tags for all the movies containing
                                                                  #the name-values in the keywords column of the movies
```

```
In [21]: """
         The method for converting the string data to list of tags, is same as that used for keywords and genres
         But in the case for crew column, the idea is to give priority to the top 4 leading acters/actresses for recommendation
         This will increase the efficiency and readability of the code(as well as the working matrix)
         This is done to get the recommendation as per the first thought that the use gets when he/she hears the name of a movie
         For example : If the user hears the name Iron Man, the first acter that will pop up in the user's mind will be
                       'Robert Downey Jr'
         """
         def top_four_people(data):                          #The data set is in string format
             List = []
             counter = 0                                     # Counter to get the top 4 crew members
             for i in ast.literal_eval(data):
                 if counter < 4:
                     List.append(i['name'])
                 counter += 1
             return List
```

```
In [22]: """
         For the case of Crew column the only need is to get the name of the director of the movie.
         People usually don't remember who was the VFX expert, or who did the final editing, or who designed the sets
         But people Do remember The Director in many cases
         For Example, the momemnt User hears the name Justice League, the first is Snyder's Cut, Which actually gives the name
                       Zack Snyder, the director of the Snyder cut

         Proving Point : What was name of the head of the vfx team?
                       : Like its mentioned, people dont remember :)

         """
         def get_me_the_director(data):
             List = []
             for i in ast.literal_eval(data):
                 if i['job'] == 'Director':
                     List.append(i['name'])
             return List
```

```
In [23]: movies['cast']= movies['cast'].apply(top_four_people)        #Provides the list of top four actors/actressses
                                                                      #for all the movies

         movies['crew']= movies['crew'].apply(get_me_the_director)    #Provides the list of directors for all the movies
```

```
In [24]: movies['overview'] = movies['overview'].apply(lambda x:x.split()) #Converts the overview string for each movie to a list
                                                                           #containg all the words in the string
```



```
In [27]: movies.head()                                    #Displaying all the changes done to refine our data
```

Out[27]:

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	[In, the, 22nd, century,, a, paraplegic, Marin...	[Action, Adventure, Fantasy, ScienceFiction]	[cultureclash, future, spacewar, spacecolony, ...	[SamWorthington, ZoeSaldana, SigourneyWeaver, ...	[JamesCameron]
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, Ste...	[GoreVerbinski]
2	206647	Spectre	[A, cryptic, message, from, Bond's, past, send...	[Action, Adventure, Crime]	[spy, basedonnovel, secretagent, sequel, mi6, ...	[DanielCraig, ChristophWaltz, LéaSeydoux, Ralp...	[SamMendes]
3	49026	The Dark Knight Rises	[Following, the, death, of, District, Attorney...	[Action, Crime, Drama, Thriller]	[dccomics, crimefighter, terrorist, secretiden...	[ChristianBale, MichaelCaine, GaryOldman, Anne...	[ChristopherNolan]
4	49529	John Carter	[John, Carter, is, a, war-weary,, former, mili...	[Action, Adventure, ScienceFiction]	[basedonnovel, mars, medallion, spacetravel, p...	[TaylorKitsch, LynnCollins, SamanthaMorton, Wi...	[AndrewStanton]

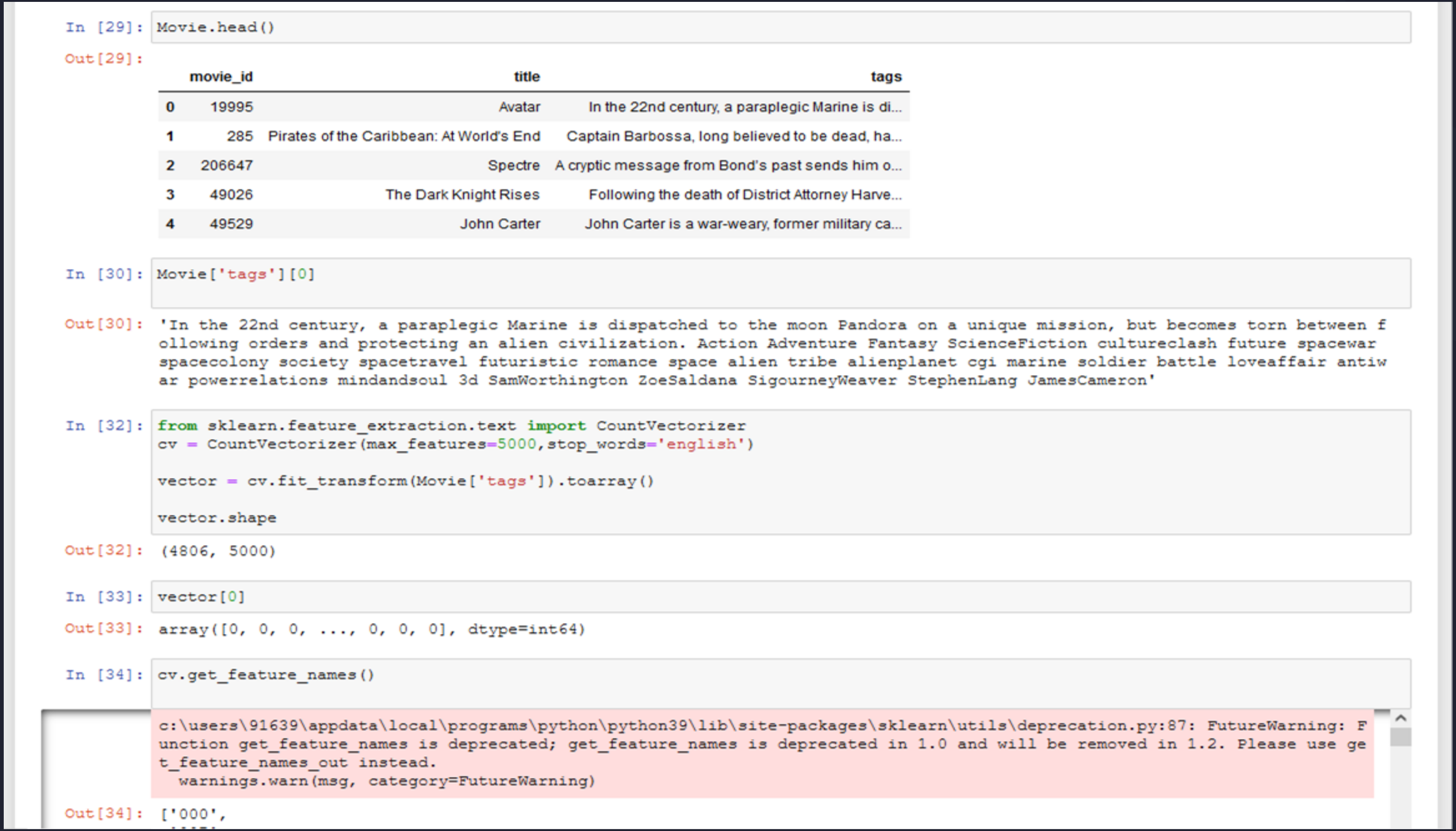
```
In [29]: movies['tags'] = movies['overview'] + movies['genres'] + movies['keywords'] + movies['cast'] + movies['crew']
```

```
In [33]: Movie = movies.drop(columns=['overview','genres','keywords','cast','crew'])
```

```
In [34]: Movie.head()
```

Out[34]:

	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...

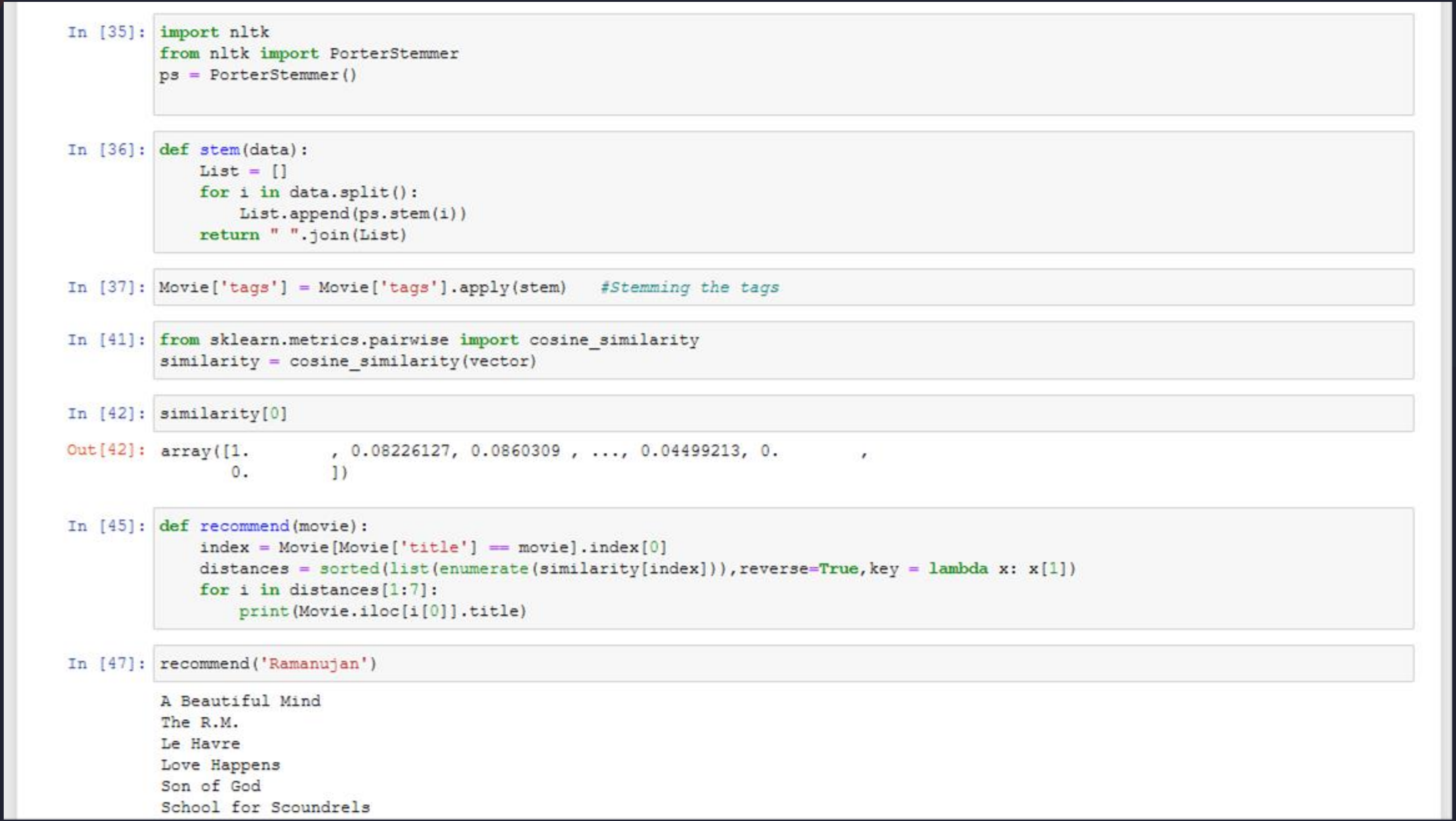


```
In [29]: Movie.head()
```

Out[29]:

	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 [30]: Movie['tags'][0]

Out[30]: 'In the 22nd century, a paraplegic Marine is dispatched to the moon Pandora on a unique mission, but becomes torn between f
         ollowing 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 antiw
         ar powerrelations mindandsoul 3d SamWorthington ZoeSaldana SigourneyWeaver StephenLang JamesCameron'

In [32]: from sklearn.feature_extraction.text import CountVectorizer
         cv = CountVectorizer(max_features=5000,stop_words='english')

         vector = cv.fit_transform(Movie['tags']).toarray()

         vector.shape

Out[32]: (4806, 5000)

In [33]: vector[0]

Out[33]: array([0, 0, 0, ..., 0, 0, 0], dtype=int64)

In [34]: cv.get_feature_names()

         c:\users\91639\appdata\local\programs\python\python39\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning: F
         unction get_feature_names is deprecated; get_feature_names is deprecated in 1.0 and will be removed in 1.2. Please use ge
         t_feature_names_out instead.
           warnings.warn(msg, category=FutureWarning)

Out[34]: ['000',
```

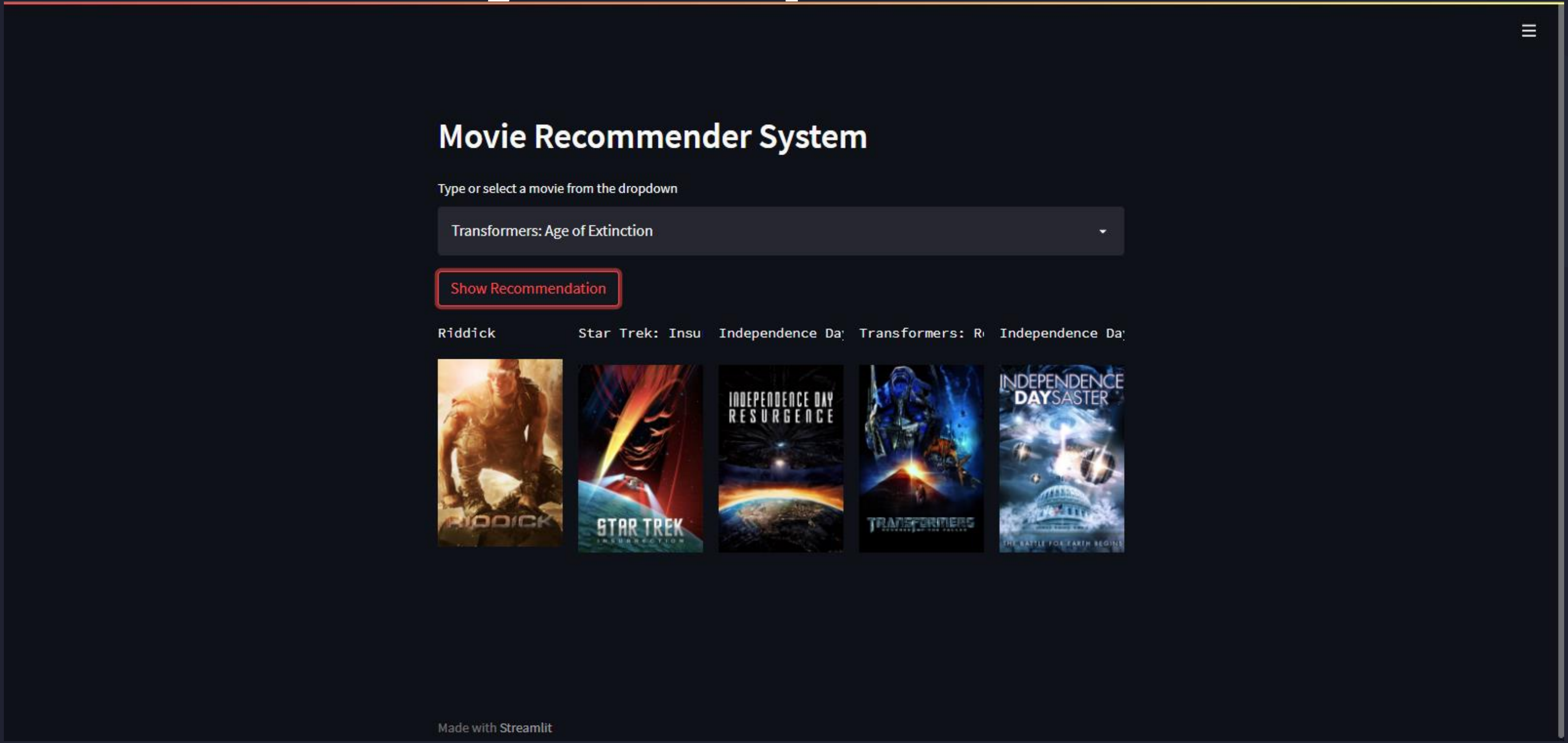


```
In [35]: import nltk
         from nltk import PorterStemmer
         ps = PorterStemmer()

In [36]: def stem(data):
             List = []
             for i in data.split():
                 List.append(ps.stem(i))
             return " ".join(List)

In [37]: Movie['tags'] = Movie['tags'].apply(stem)    #Stemming the tags

In [41]: from sklearn.metrics.pairwise import cosine_similarity
         similarity = cosine_similarity(vector)

In [42]: similarity[0]

Out[42]: array([1.        , 0.08226127, 0.0860309 , ..., 0.04499213, 0.        ,
                0.        ])

In [45]: def recommend(movie):
             index = Movie[Movie['title'] == movie].index[0]
             distances = sorted(list(enumerate(similarity[index])),reverse=True,key = lambda x: x[1])
             for i in distances[1:7]:
                 print(Movie.iloc[i[0]].title)

In [47]: recommend('Ramanujan')

         A Beautiful Mind
         The R.M.
         Le Havre
         Love Happens
         Son of God
         School for Scoundrels
```

References

- https://scikit-learn.org/stable/
- https://pandas.pydata.org/
- https://docs.python.org/3/library/ast.html
- https://numpy.org/
- https://www.nltk.org/
- https://docs.python.org/3/library/pickle.html
- https://www.upwork.com/resources/what-is-content-based-filtering

Result (also disused in demo video): Actual Project Snapshot



Movie Recommender System

Type or select a movie from the dropdown

Transformers: Age of Extinction

Show Recommendation

Riddick

Star Trek: Insu

Independence Da

Transformers: R

Independence Da

Made with Streamlit

Demo Video (Includes result & conclusion)

Conclusion

Movie Recommender System is a system where anyone can write the name of the movie and related to the name, user will get recommended similar movies using content-based filtering. It is created in an easy implementation environment and provides the developer the flexibility to modify the system according to the requirements in the future.

THANKYOU