

SUMMER INTERNSHIP PROJECT REPORT
ON

“MOVIE RECOMMENDATION ENGINE”

submitted in partial fulfilment of the requirements for the award of the degree of

B.TECH (CSE)

SUBMITTED BY:

Yashvardhan Rana

A51405218009

Under the Guidance of

Dr. Rajesh Kumar Tyagi

Assistant Professor

ASET

Dr. Rashmi

Assistant Professor

ASET



Department of Computer Science & Engineering

Amity School of Engineering & Technology

AMITY UNIVERSITY HARYANA

October 2020

Table of Content

Content	Page no.
Declaration	4
Certificate	5
Acknowledgement	6
Course Certificate	7
Abstract	8
List of Figures	9
Chapter I	
1. Introduction	10
Different Types of Recommender System	11
2. Objective	12
Chapter II	
2.1. Tools and Platform	13
2.2. Scope of project	14
2.3. Technology and Library	15
2.3.1. Software	15
2.3.2. Libraries	16
Chapter III	
3.1. Research Methodology	17
3.2. Dataset	18
3.3. Walkthrough of building a recommender system	
3.3.1. Data Collection	19
3.3.2. Data Processing	20
3.3.3. Machine Learning Model	23
Method	24
3.3.4. Website	28

Chapter IV	
Conclusion	29
Chapter V	
Suggestions	30
References	31



Amity School of Engineering & Technology

Declaration

I, Yashvardhan Rana having Enrolment Number - A51405218009, is student of Bachelor of Technology in Department of Information Technology, Amity School of Engineering and Technology, Amity University Haryana, hereby declare that we are fully responsible for the information and results provided in this project report titled “Movie Recommendation Engine” submitted to Department of Computer Science and Engineering, Amity School of Engineering and Technology, Amity University Haryana. We have taken care of all aspects to honour the intellectual property rights and have acknowledged the contributions of others for using them. Our supervisor, Head of the department and the Institute should not be held for full or partial violation of copyrights if found at any stage of our degree.

Signature

Yashvardhan Rana

Enrolment Number

A51405218009



Amity School of Engineering & Technology

Certificate

This is to certify that the work in the project report entitled “Movie Recommendation System” by Yashvardhan Rana bearing Enrolment Number – A51405218009, is a Bonafede record of project work carried out by them under my supervision and guidance in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Information Technology in the Department of Computer Science and Engineering, Amity School of Engineering and Technology, Amity University Haryana, Gurgaon. Neither this project nor any part of it has been submitted for any degree or academic award elsewhere.

Signature of Supervisor

Date:

Dr. Rajesh Kumar Tyagi

Assistant Professor

ASET

Signature of Co-Supervisor

Date:

Mr. Rashmi

Assistant Professor

ASET



Amity School of Engineering & Technology

ACKNOWLEDGMENT

Acknowledgement is not a mere obligation but the epitome of humility and ineptness to all those who have helped in the completion of this project. I am thankful to Mr. Anshul Porwal, Digital India Corporation for his constant guidance and encouragement provided in this endeavour. I also thank my parents for their continued support, understanding and patience without whose support and understanding this endeavour would never be fruitful. I also thank all my friends who helped me out in completing this project and helping me to solve various problems encountered during the progress of this project.

Yashvardhan Rana

Enrolment Number

A51405218009

ABSTRACT

A recommendation engine filters the data using different algorithms and recommends top 5 most relevant movies to the user. It first takes the input of the customer and based on that, recommends movies which the user might like to watch. The engine selects the movies from a given dataset which contains details of 5000 movies and recommends other movies that have similar description, cast, genre, etc. The dataset can be edited and the content of the dataset can be edited to increase the number of movies to provide better recommendations. This is a project which describes Content Based Recommender Engine.

The project has been implemented using Python programming language. Pycharm and Jupyter software has been used to run the code. Libraries that have been used for the project have been explained further which include pandas, numpy, ast, sci-kit, etc.

INTERNSHIP CERTIFICATE




Date: 15th July 2021

TO WHOM IT MAY CONCERN

This is to certify that Mr. Yashvardhan Rana S/O` MR. Anuj Rana, student of bachelors of Technology (B. Tech) - Computer Science & Engineering, Amity University, and Haryana has successfully completed Internship on **Data Science with Python and Machine Learning** from 4th June 2021 - 5th July 2021 from Digital India Corporation under the guidance of Mr. Anshul Porwal.

We have found him to be a self starter who is motivated, duty bound, hard-working and punctual. He has worked sincerely in his project and his performance was par excellence.

We wish him success in his life and career.



(Gaurav Sharma)
Principal Research Scientist &
Officer-in-Charge (HR)

List of Figures

Figures	Page no.
Figure1: Engine Page	17
Figure2: Importing main libraries	18
Figure3: Movie dataset	20
Figure4: Credits dataset	20
Figure5: Dataset after extraction	21
Figure6: Genres in list format	21
Figure7: List column in movie dataset	22
Figure 8: Tags column	23
Figure 9: Movie Dataframe	23
Figure 10: Stemming	24
Figure 11: Vectorization	25
Figure 12: Similarity function	26
Figure 13: Recommendation function	27
Figure 14: Pickle function	27
Figure 14: PyCharm Code	28

Chapter I

Introduction

What is a recommender system?

A recommender system is a simple algorithm whose aim is to provide the most relevant information to a user by discovering patterns in a dataset. The algorithm rates the items and shows the user the items that they would rate highly. An example of recommendation in action is when you visit Amazon and you notice that some items are being recommended to you or when Netflix recommends certain movies to you. They are also used by Music streaming applications such as Spotify and YouTube Music to recommend music that you might like.

Below is a very simple illustration of how recommender systems work in the context of an e-commerce site.

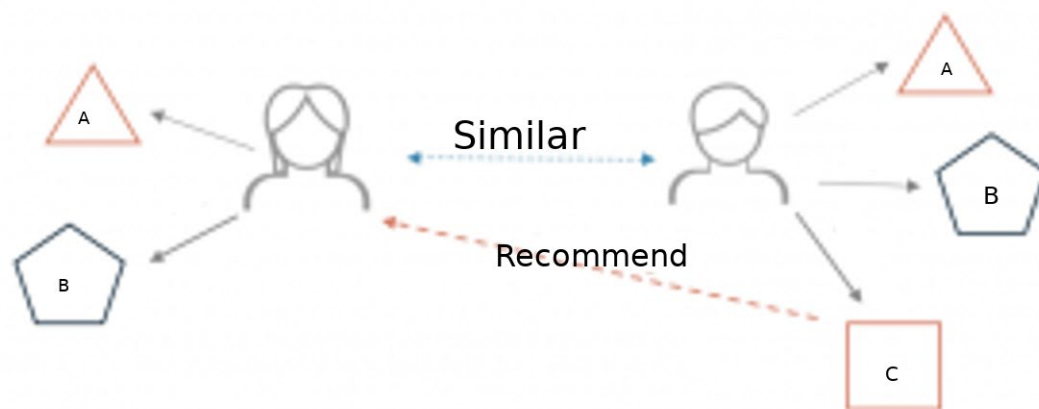


Figure: Recommender System Example

Different types of recommendation engines

The most common types of recommendation systems are **content-based** and **collaborative filtering** recommender systems. In collaborative filtering, the behaviour of a group of users is used to make recommendations to other users. The recommendation is based on the preference of other users. A simple example would be recommending a movie to a user based on the fact that their friend liked the movie. There are two types of collaborative models **Memory-based** methods and **Model-based** methods. The advantage of memory-based techniques is that they are simple to implement and the resulting recommendations are often easy to explain. They are divided into two:

- User-based collaborative filtering: In this model, products are recommended to a user based on the fact that the products have been liked by users similar to the user. For example, if Derrick and Dennis like the same movies and a new movie come out that Derrick like, then we can recommend that movie to Dennis because Derrick and Dennis seem to like the same movies.
- Item-based collaborative filtering: These systems identify similar items based on users' previous ratings. For example, if users A, B, and C gave a 5-star rating to books X and Y then when a user D buys book Y they also get a recommendation to purchase book X because the system identifies book X and Y as similar based on the ratings of users A, B, and C.

Content-based systems use metadata such as genre, producer, actor, musician to recommend items say movies or music. Such a recommendation would be for instance recommending Infinity War that featured Vin Diesel because someone watched and liked The Fate of the Furious. Similarly, you can get music recommendations from certain artists because you liked their music. Content-based systems are based on the idea that if you liked a certain item you are most likely to like something that is similar to it.

Objective

The use of online platforms for purchasing items, watching content and listening to music and content is on rise. Creating a recommender engine helps the provider to provide similar and related items or content that the user might like to consume. This also saves time and increase the ease to find what the user is looking for.

I learnt python to understand coding language in a better way and it is said that python is of the easiest and most comfortable coding language to start with. It is a user-friendly language and a user can learn this language quickly. Movie Recommendation Engine is one of my first Machine Learning projects on python. Python is a language that allows you to run the code on many other software which makes it easy to understand and complete work with more speed.

Other software's like sublime text editor, PyCharm and Anaconda are really common software's which helps us to understand the code and also suggests us more functions. They also allow us to check errors and helps to rectify them. We have learned and used Content-based recommender system for the project.

Goal of the Recommendation engine is to provide user accurate and similar results that the user might like. As discussed before, it saves time and gives the user better experience in searching for the item or content he/she would be looking for.

Chapter II

1.1. Tools and Platform

Minimum requirement:

Processor:	Intel Atom® processor or Intel® Core™ i3 processor
CPU Speed:	1.4 GHz
RAM:	512 MB
Hard Disk:	80 GB
OS:	Windows* 7 or later, macOS, and Linux
Mouse:	Yes
Keyboard:	Yes
Sound Card:	NO
Language:	Python* versions: 2.7.X, 3.6.X

2.2 Scope of Project

The objectives of the proposed Project is to use machine learning and making an algorithm to select similar or related data that has to be given to the user from a very big data or dataset that the user might like or is looking for.

The engine takes a movie from the user as an input and returns top 5 similar or related movies the user might like to watch on the basis of cast, director, description, genre, etc.

It is manually a very difficult job to perform such kind of a search where in this case, the engine searches for the best recommendation to the user which saves lots of time and energy of the user.

2.3. Technology and Library Used

2.3.1. Software

1. Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

2. Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

3. PyCharm

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains (formerly known as IntelliJ). It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda.

1.3.2. Libraries

I. Pandas

Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc. In this tutorial, we will learn the various features of Python Pandas and how to use them in practice.

II. AST (abstract syntax tree)

Provides tree structure for your source code (Python or any other programming language). In Python, AST is compiled into a code object using the built-in “compile()” function.

III. SciKits

SciKits (short for SciPy Toolkits) are add-on packages for SciPy, hosted and developed separately and independently from the main SciPy distribution. All SciKits are licensed under OSI-approved licenses.

IV. Natural Language Toolkit

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

V. 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.

Chapter III

3.1. Research and Methodology

Project

The name of the project says it all, which is Movie Recommendation Engine. It is a Content Based Recommendation Engine which is coded in Python programming language. The program has the feature to ask a movie from the user as an input and the program searches for similar movies based on the content of the movie provided.

After the input is provided by the user, it proceeds to searching for similar movies in a dataset which contains 5000 movies data. The dataset contain data related to the movie description, cast, crew, director, movie ID, title and genres.

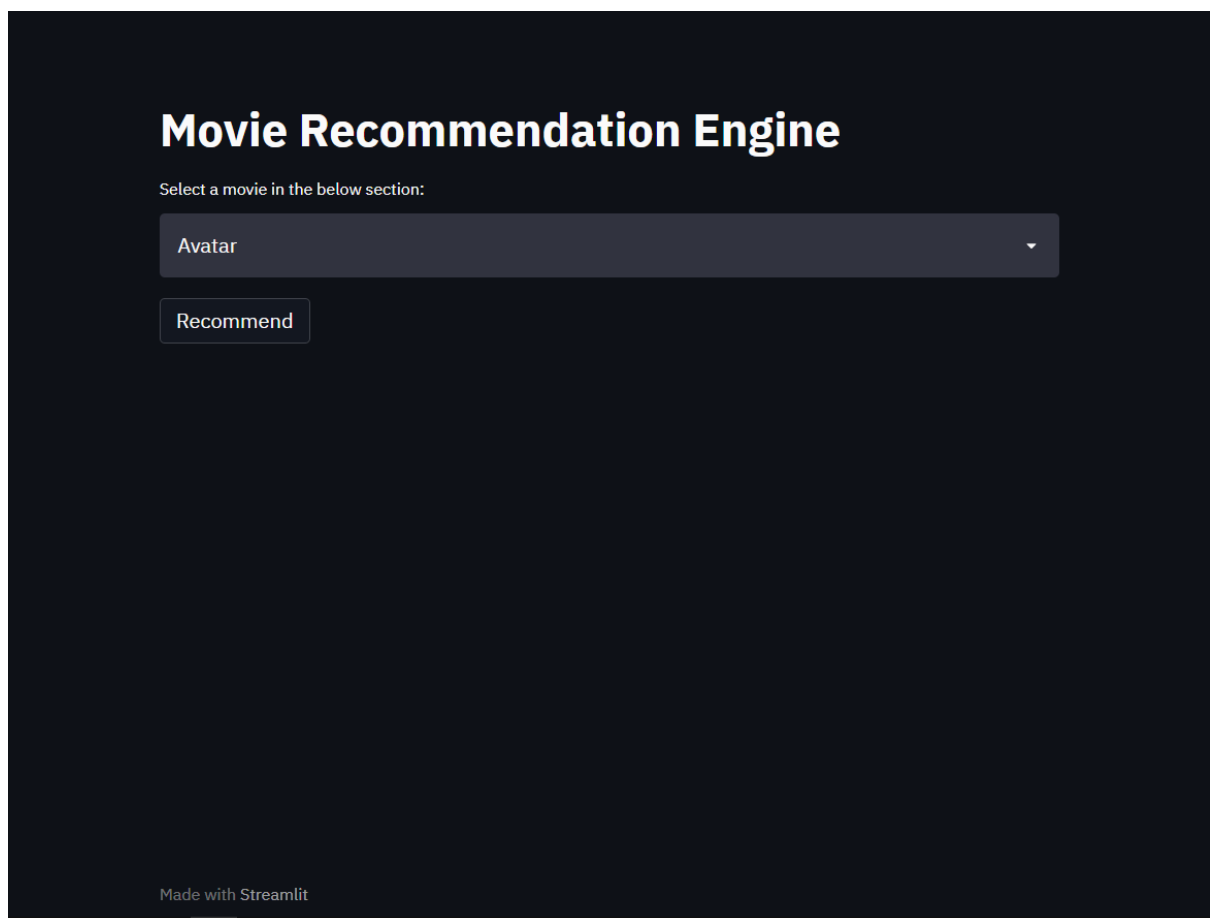


Figure1: Engine Page

The engine works on a local host webpage. The user is suppose to select form the given list or can search for the movie that he/she like and click on the recommend button. After that the engine will search for the movies have similarities and append those movies to a list. After the process the system will recommend top 5 movies that are similar to the input movie. The user can select from the recommend movies and have no hassle to search manually.

3.2. Dataset used for building this recommender system

In this project, we have taken a dataset from Kaggle (<https://www.kaggle.com/tmdb/tmdb-movie-metadata>) which contains two files:

1. Movies: contains all the data about the movie- genre, movie ID, budget, release date, etc.
2. Credits: contains the data about the cast, crew, movie ID and title.

The dataset contains the data of 5000 movies.

3.3. Walkthrough of building a recommender system

1. Data collecting
2. Processing
3. Machine Learning model
4. Methodology
5. Website

3.3.1. Data collecting

Any content based recommender system starts with selection of the dataset that have enough data about the item that the machine is going to recommend.

Mentioned above, for this project we have select a dataset that contains data of 5000 movies and have more than enough categories to compare each other and find the similar movies that would suit the user.

After this step, we have used Jupyter notebook for analysing the working on the dataset with the help of Pandas and Numpy libraries.

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

[2]: movies = pd.read_csv('D:/Projects/Recommendation Engine (ML)/Movie Engine/TMDB 5000 Movie Dataset/Movies.csv')
      credits = pd.read_csv('D:/Projects/Recommendation Engine (ML)/Movie Engine/TMDB 5000 Movie Dataset/Credits.csv')
```

Figure 2: Importing main libraries

3.3.2. Data processing

Data processing is the process in which we clean the data that we have, by selecting the columns that we need and removing the unwanted data that will not be used in the project. We also check and remove all the null values which are not required in the project. In simple language, cleaning the data to make it easier to read and process for a particular project.

[3]:

movies.head()

[3]:	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies	production_countries	release_date	revenue	runtime
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id": 12, "name": "Avatar"}]	en	Avatar	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 ancient civilization.	150.437577	[{"name": "Ingenious Film Partners", "id": 289}, {"name": "Twentieth Century Fox", "id": 1}], [{"iso_3166_1": "US", "name": "United States of America"}]	2009-12-10	2787965087	1	
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}, {"id": 28, "name": "Action"}]	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "name": "pirates"}, {"id": 12, "name": "Adventure"}]	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, has returned to the Caribbean Sea.	139.082615	[{"name": "Walt Disney Pictures", "id": 2}, {"name": "Paramount Pictures", "id": 1}], [{"iso_3166_1": "US", "name": "United States of America"}]	2007-05-19	961000000	1	
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://www.sonyictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name": "bond"}, {"id": 12, "name": "Adventure"}]	en	Spectre	A cryptic message from Bond's past sends him on a new mission.	107.376788	[{"name": "Columbia Pictures", "id": 5}, {"name": "United Kingdom", "id": 1}], [{"iso_3166_1": "GB", "name": "United Kingdom"}]	2015-10-26	880674609	1	
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "name": "Drama"}, {"id": 12, "name": "Adventure"}]	http://www.thedarkknighttrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853, "name": "dark knight"}, {"id": 12, "name": "Adventure"}]	en	The Dark Knight Rises	Following the death of District Attorney Harvey Dent, Batman deduces a link between the mobster who financed Dent's campaign and the mobster who financed the attack on Gotham City.	112.312950	[{"name": "Legendary Pictures", "id": 923}, {"name": "Warner Bros. Entertainment", "id": 1}], [{"iso_3166_1": "US", "name": "United States of America"}]	2012-07-16	1084939099	1	

Figure3: Movies data file

```
[4]: credits.head()
```

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "credit_id": "52fe48009251416c750aca23", "de...}	[{"credit_id": "52fe48009251416c750aca23", "de...}
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Sparrow", "credit_id": "52fe4232c3a36847f800b579", "de...}	[{"credit_id": "52fe4232c3a36847f800b579", "de...}
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "credit_id": "54805967c3a36829b5002c41", "de...}	[{"credit_id": "54805967c3a36829b5002c41", "de...}
3	49026	The Dark Knight Rises	[{"cast_id": 2, "character": "Bruce Wayne / Batman", "credit_id": "52fe4781c3a36847f81398c3", "de...}	[{"credit_id": "52fe4781c3a36847f81398c3", "de...}
4	49529	John Carter	[{"cast_id": 5, "character": "John Carter", "credit_id": "52fe479ac3a36847f81398c3", "de...}	[{"credit_id": "52fe479ac3a36847f81398c3", "de...}

Figure4: Credits data file

The movies data file that we have import contains a lot of data columns that were not needed in this project. We have merged the two datasets first to make it a single dataset named as 'movies'. After merging those datasets, we select the columns that we needed for the project and other unwanted columns were removed. The columns that we selected for the recommendation engine out of 23 provided columns are:

1. Movie ID
2. Title
3. Overview
4. Genres
5. Keywords
6. Cast
7. Crew

Columns ¶

Genres, ID, Keywords, Title, Overview, Cast, Crew

```
[11]: movies = movies[['movie_id', 'title', 'overview', 'genres', 'keywords', 'cast', 'crew']]
[12]: movies.head()
```

	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...

Figure5: Dataset after extracting

After processing, cleaning and sorting the data into a proper form and checking and dropping the null values. We convert all the columns from string to list form to use it together with other columns.

For the conversion we have used 'ast' library. `ast.literal_eval()` is the function used to convert string to list format. we created a function using `ast.literal_eval()` to convert all the string columns to list columns: Genre, overview, keywords, cast.

```
[17]: import ast #To convert string into List
[18]: def convert(obj):
      L = []
      for i in ast.literal_eval(obj):
          L.append(i['name'])
      return L
[19]: movies['genres'] = movies['genres'].apply(convert)
[20]: movies.head()
```

	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...

Figure6: Genres in list form

In the cleaning process of the dataset the columns were also sorted and cleaned for getting better results and for the process to take less time to provide the results.

Only the top 3 actors are extracted from the cast column and directed is selected from the crew column. This processing is also done with the help of 'ast' library.

```
[30]: movies['overview'] = movies['overview'].apply(lambda x:x.split())
```

```
[31]: movies.head()
```

	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 ...	[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]

Figure7: List columns in the movie dataset.

The next step is to model the data according to the program requirement using other tools in python.

3.3.3. Machine Learning Modelling

In machine learning modelling we have to use the data to complete our purpose and provide correct results from the dataset.

The very first step that we have done is removed all the spaces in between the words to reduce confusion in system and increase accuracy and better results. Then we have created a new column named 'Tags' which contain all the words from: overview, genres, keywords, cast and crew.

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

```
[34]: movies.head()
```

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

Figure8: Tags column

A new DataFrame is created 'moviesdf' in which only contains 3 columns which are: MovieID, title, tags. Tags column has been converted from list format to string format so that we can use to data to extract words and use it compare. Here 'tags' is the column that contains the data after concatenating other 5 columns. This makes the dataset look clean and easy to understand and work with.

```
moviesdf.head()
```

	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...

Figure9: Moviesdf dataframe

Methodology

The method we are going to use in the project is content based. For which we have created the 'Tags' columns that contains all the data related to the specific movie. To reduce the occurrence of same kind of words, we have used nltk library to stem the words. All the characters in the tag column are convert into lower format.

```
: # Library and function for stemming words

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)

#command for stemming all the words in tag

moviesdf['tags'] = moviesdf['tags'].apply(stem)

<ipython-input-64-289911b5e5e1>: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
moviesdf['tags'] = moviesdf['tags'].apply(stem)
```

Figure10: Stemming of words

In this technique we convert the whole text into vectors (Text of Tags column) for finding similarities in the text of movie description. From this process, scikit-learn library is used. By using sklearn which is a function of scikit library, we were able to do the vectorization process.


```

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

: cv.fit_transform(moviesdf['tags']).toarray()

: 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)

: cv.fit_transform(moviesdf['tags']).toarray().shape

: (4806, 5000)

: vectors= cv.fit_transform(moviesdf['tags']).toarray()

: vectors

: 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)

```

Figure11: Vectorization

Now as we have 5000 vectors (each movie is now a vector). The techniques we have used in this system is ‘Bag of Words’ technique. In this technique we combine all the words and then extract the most common occurring words. Now we have 5000 words that are most common in all the tags and 5000 movies, we create an array for the same.

The method we use in this project is that we compare movie tags with the 5000 common word text and then select the closest 5 vectors and recommend them to the user.

To find the similarity we have used the `cosine_similarity` library to find the cosine distance which is the degree difference between the vectors. The closest vectors according to cosine distance are displayed to the user. A recommender function is created to sort and fetch the top 5 indexes.

```
: from sklearn.metrics.pairwise import cosine_similarity
:
: similarity= cosine_similarity(vectors)
:
: cosine_similarity(vectors).shape
:
: (4806, 4806)
:
: similarity[0]
:
: array([[1.          , 0.08458258, 0.08718573, ..., 0.04559608, 0.
:         0.          ]])
:
: sorted(similarity[0],reverse=True)
:
: [0.9999999999999998,
:  0.29061909685954823,
:  0.26401000024165,
:  0.25903973506580724,
:  0.2537477434955704,
:  0.2507061052819501,
:  0.2484013136974297,
:  0.24784079854830487,
:  0.23995690956687135,
:  0.23490461932490855,
:  0.23485569615051044,
:  0.23089735286521348,
:  0.22830583024432843,
:  0.22147019900891643,
:  0.21524880100025257,
:  0.21369687880543226,
:  0.213352298825066,
:  0.2092457497388747,
:  0.20712325533373332,
```

Figure12: Similarity function

Enumerate function is used to find the distance from one movie to another.

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

[(1216, 0.29061909685954823),
 (3730, 0.26401000024165),
 (507, 0.25903973506580724),
 (539, 0.2537477434955704),
 (2409, 0.2507061052819501)]

def recommend(movie):
    movie_index = moviesdf[moviesdf['title'] == movie].index[0]
    distances = similarity[movie_index]
    movies_list= sorted(list(enumerate(similarity[0])),reverse=True, key= lambda x:x[1])[1:6]

    for i in movies_list:
        print(moviesdf.iloc[i[0]].title)

recommend('Avatar')

Aliens vs Predator: Requiem
Falcon Rising
Independence Day
Titan A.E.
Aliens
```

Figure13: Recommendation function

Lastly Pickly library is used to dump and get the files created so that we can use them in our PyCharm editor to run the code and use it through a website.

```
: import pickle

: pickle.dump(moviesdf, open('movies.pkl','wb'))

: moviesdf['title'].values

: array(['Avatar', "Pirates of the Caribbean: At World's End", 'Spectre',
: ..., 'Signed, Sealed, Delivered', 'Shanghai Calling',
: 'My Date with Drew'], dtype=object)

: pickle.dump(moviesdf.to_dict(), open('movie_dict.pkl','wb'))

: pickle.dump(similarity,open('similarity.pkl','wb'))
```

Figure14: Pickle function

3.3.4. Website

The website for this recommendation engine is running on localhost. Streamlit library is used to create the website. Pickle is used for the dataframe used from the jupyter notebook.

This program takes the input, compares with other movies and provides the output.

```
import streamlit as st
import pickle
import pandas as pd

def recommend(movie):
    movie_index = movies[movies['title'] == movie].index[0]
    distances = similarity[movie_index]
    movies_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[1:6]

    recommended_movies = []
    for i in movies_list:
        recommended_movies.append(movies.iloc[i[0]].title)
    return recommended_movies

movies_dict = pickle.load(open('movie_dict.pkl', 'rb'))
movies = pd.DataFrame(movies_dict)

similarity = pickle.load(open('similarity.pkl', 'rb'))
st.title('Movie Recommendation Engine')

selected_movie_name = st.selectbox(
    'Select a movie in the below section: ',
    movies['title'].values
)

if st.button('Recommend'):
    recommendations = recommend(selected_movie_name)
    for i in recommendations:
        st.write(i)
```

Figure15: Pycharm code

Chapter VI

Conclusion

Achievements

Through creating this project, we were able to learn about many new libraries and their functions. Now we understand that how we clean, process the data in a dataset and use it for analysis and sorting. Many other machine learning aspects that we have learned in this project will help us in future projects. We have also concurred problem solving skills and how to bring out the best possible results.

Chapter V

Suggestions for Future Improvements

This system can be improved by building a Memory-Based Collaborative Filtering based system. In this case, we'd divide the data into a training set and a test set. An alternative is to build a Model-based Collaborative Filtering system. This is based on matrix factorization. Matrix factorization is good at dealing with scalability and sparsity than the former. You can then evaluate your model using techniques such as Root Mean Squared Error (RMSE). Also creating a website and establishing this for actual users will be a great improvement in this project.

References

1. <https://www.datacamp.com/community/tutorials/recommender-systems-python>
2. https://pandas.pydata.org/pandas-docs/stable/user_guide/index.html
3. https://scikit-learn.org/stable/user_guide.html
4. <https://docs.python.org/3/library/ast.html>
5. <https://www.codeproject.com/Articles/1251123/Building-Recommendation-Systems-using-Python>
6. <https://docs.python.org/3/library/pickle.html>
7. <https://medium.com/analytics-vidhya/content-based-recommender-systems-in-python-2b330e01eb80>