MOVIE RECOMMENDATION SYSTEM USING CONTENT-BASED FILTERING

A Project Report is submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Submitted by

CHERUKURI VISHAL DEEP, 121910313021

JANGA NAGA VENKATA SAI HARIDEEP, 121910313024

ALLU TATAJI, 121910313060

Under the guidance of

DECLARATION

I/We hereby declare that the project report entitled "MOVIE RECOMMENDATION SYSTEM USING CONTENT BASED FILTERING" is an original work done in the Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of the requirements for the award of the degree of B.Tech. in Computer Science and Engineering. The work has

not been submitted to any other college or university for the recognition of any degree or diploma

Date: 26-10-2022

Registration NO(s).Name(s) Signature(s)

121910313021 VISHALDEEP CHERUKURI

121910313024 HARIDEEP JANGA

121910313060 ALLU TATAJI

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING GITAM SCHOOL OF TECHNOLOGY

GITAM

+

CERTIFICATE

This is to certify that the project report entitled "MOVIE RECOMMENDATION SYSTEM USING CONTENT BASED FILTERING" is a bona fide record of work carried out by CHERUKURI VISHALDEEP (121910313021), JANGA NAGA VENKATA SAI HARIDEEP(121910313024), ALLU TATAJI (121910313060) students submitted in partial fulfillment of the requirements for the award of the degree of Bachelors of Technology in Computer Science and Engineering.

TABLE OF CONTENTS **PAGE NO**

ABSTRACT

A Movie recommendation system is a system that provides suggestions to users using content-based filtering, which recommends movies based on the content of the movies.
Movie recommendation systems aim at helping movie enthusiasts by suggesting what movie to watch without having to go through the long process of choosing from a large set of movies that go up to thousands and millions which is time-consuming and confusing.

This project develops a Movie Recommendation System to recommend movies based on different parameters. The principal objective of the project is to construct a movie recommendation framework to prescribe pictures to users. For this model, we are importing NumPy, pandas, and pickle and downloaded a dataset from Kaggle. After preprocessing the dataset and combining all the attributes into one attribute, the data is vectorized using vectorization and fed to cosine similarity which finds the cosine distance between the given vectors and finds the similarity between them. The recommendation system takes one movie as input from the given data set and outputs 5 movies as a recommendation result. A web-based platform is built in a python programming language using streamlit . These systems which are based on content recommendation are mannered to people, these systems do not recommend anything to the user, it limits your choice.

INTRODUCTION

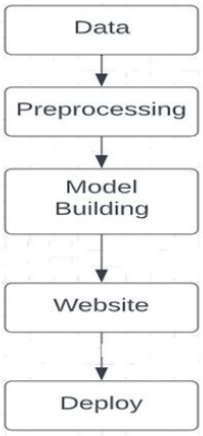
This recommendation system has come to play a vital role in Netflix, Amazon, and Facebook where the system recommends based on user preferences, this system is useful to recommend based on the content, when you search for a movie in the search tab it recommends a movie based on the content of the searched movie. Assume that you have searched for mobile in the Amazon and Flipkart applications then, when you open an app after some time it will show you the mobile phones that were related to your previous search history similarly it will appear on Amazon, and Netflix where the system recommends a movie based on the previous search history and movie ratings these websites recommend based on watching history and ratings and that will help you in suggesting the movies that you would like to watch.

These recommendations are made by filtering techniques. There are three types of recommendation techniques that are Content-based filtering, Collaborative filtering, and Hybrid-based recommendation. In this project, I used content-based filtering, I had taken a dataset and I had imported libraries like NumPy, pandas, and pickle, and for building a website I used streamlit. The main goal of this project is to build a model which helps the user in an easy recommend movies by the system based on the user inputs.

LITERATURE REVIEW

Planning a literature review and preparing a position paper before beginning is essential to comprehending AI-based algorithms. A proposal framework, often known as a suggestion engine, is a model used for data sifting that aims to predict a client's preferences and provide recommendations based on those preferences. Two main user kinds are dealt with in any recommendation engine. They are both active and recent users. Users that have rated a few films and are active do so.

SYSTEM METHODOLOGY



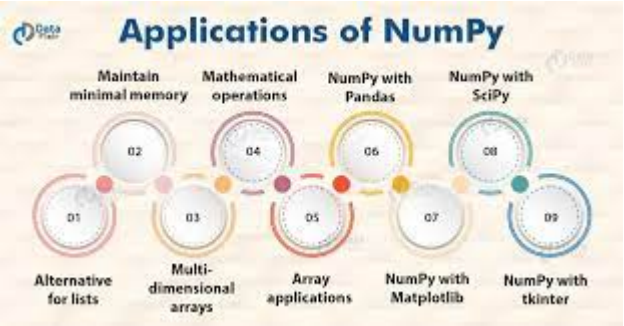
Firstly we have to import a specific dataset and libraries then we have to preprocess the dataset removing all the null values and then we have to combine all the selected attributes into a single feature. We'll need to convert the text data from that feature into vectors by vectorization.

Later, we must determine the similarity score of the vectors, then finally obtain a recommendation based on the system architecture described below. Firstly, I collected the two datasets from the kaggle and imported them into jupyter notebook then took sufficient libraries will read the data set then, took the necessary attributes to produce a similarity score and combined the datasets with these attributes, and processed the datasets for removing null values. Then consider the above attributes to convert the text data into vectors by using a vectorization tool these functions will convert the text data into vectors in a 2D matrix and use a technique called "Bag of Words" to combine all the similar words in one bag. Then by calculating the cosine distance between the vectors to generate the similarity score by using the 'cosine similarity tool' a recommended method is built using cosine-similarity and count vectorizer.

OVERVIEWS OF TECHNOLOGIES

NUMPY

Working with arrays is made possible by the Python package NumPy. Additionally, it provides functions for working with matrices, the Fourier transform, and the linear algebra domain Travis Oliphant developed NumPy in 2005. You may use it without restriction as it is an open-source project. Numerical Python is known as NumPy.



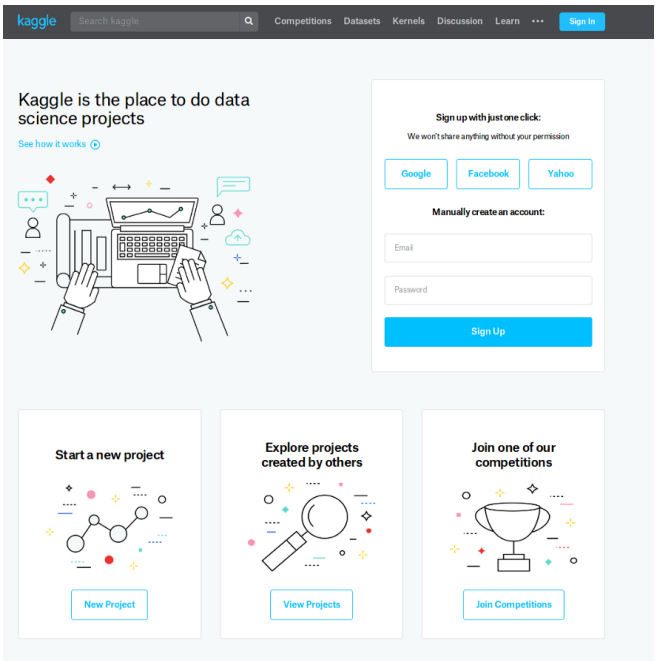
PANDAS

Pandas is an open-source library designed primarily for working quickly and logically with relational or labeled data. It offers a range of data structures and procedures for working with time series and numerical data. The NumPy library serves as the foundation for this library. Pandas is quick and offers its users exceptional performance & productivity.

STREAMLIT

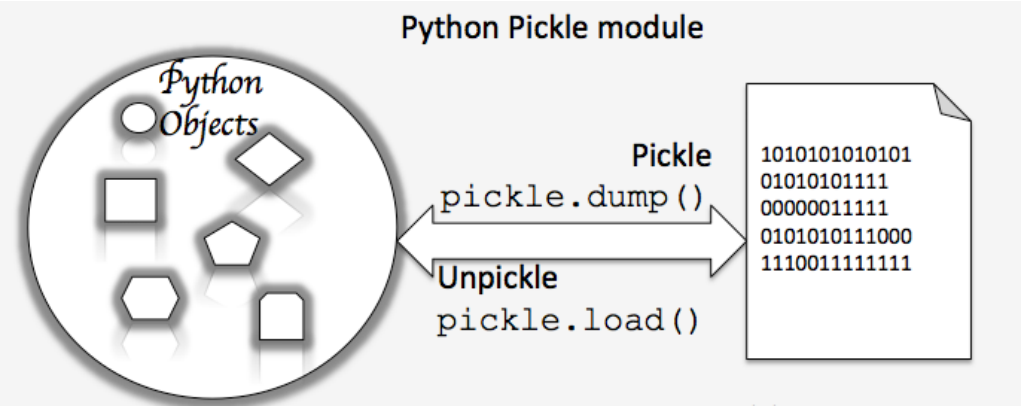
Streamlit is a free and open-source framework for quickly creating and sharing visually appealing machine learning and data science web apps. It is a Python-based library created specifically for machine learning engineers. Data scientists and machine learning engineers are not web developers, and they are not interested in spending weeks learning how to build web apps using these frameworks. They prefer a simpler tool to learn and use, as long as it can display data and collect necessary parameters for modeling. With Streamlight, you can create a visually stunning application with just a few lines of code.

KAGGLE



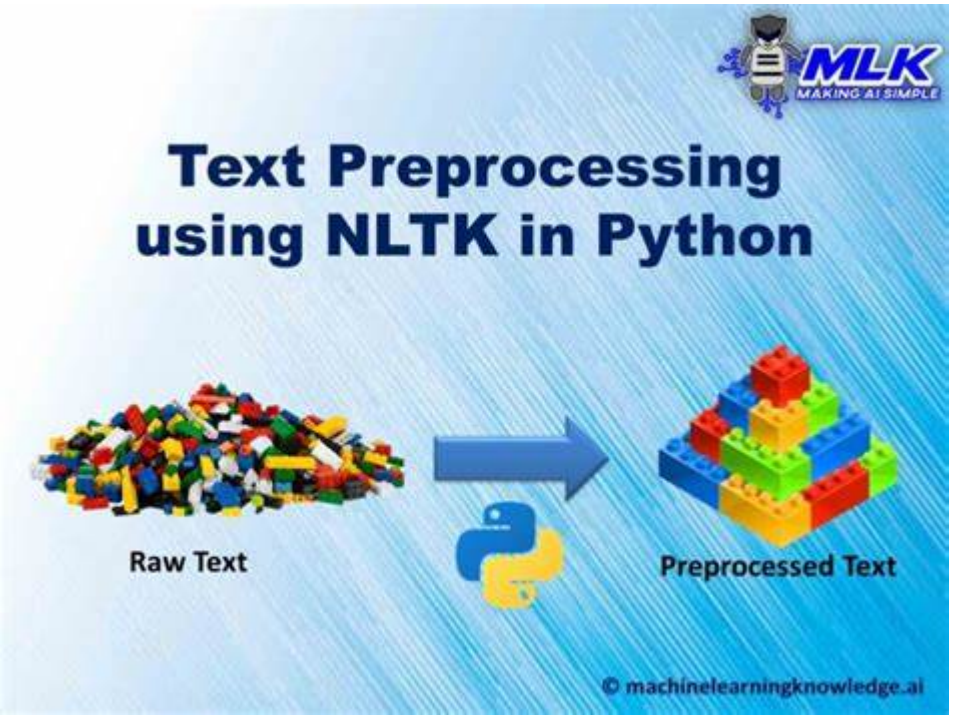
Kaggle, a subsidiary of Google LLC, is an online community of data scientists and machine learning practitioners. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.

PICKLE



The pickle module is used for implementing binary protocols for serializing and deserializing a Python object structure. Pickling is the process of converting a Python object hierarchy into a byte stream. Pickling's inverse is unpickling, which converts a byte stream into an object hierarchy.

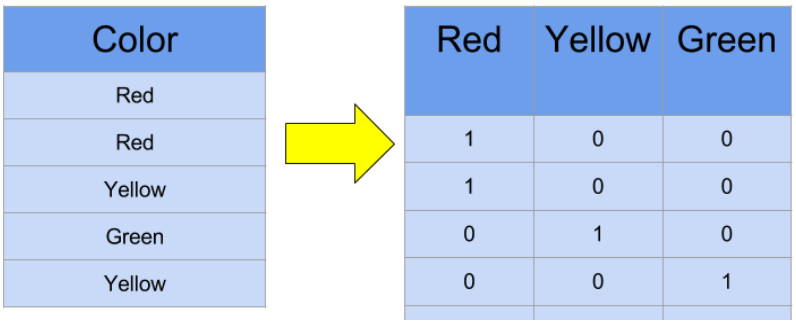
NLTK



Natural language processing (NLP) is the study of how to make natural human language understandable to computer programs. Natural Language Toolkit (NLTK) is a Python package that can be used for NLP. A large portion of the data you may be analyzing is unstructured and contains the human-readable text. You must first preprocess the data before you can analyze it programmatically. In this tutorial, you'll get your first taste of the types of text preprocessing tasks you can perform with NLTK, so you'll be

prepared to use them in future projects. You'll also learn how to perform basic text analysis and create visuals.

COUNT VECTORIZER

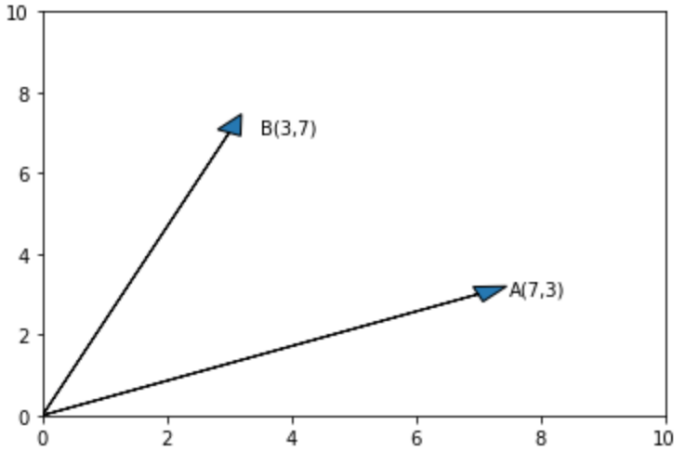


Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

To use textual data for predictive modeling, the text must be parsed to remove specific words a process known as tokenization. These words must then be encoded as integers or floating-point values before they can be used as inputs in machine learning algorithms. This is known as feature extraction (or vectorization). The CountVectorizer in Scikit-learn is used to convert a set of text documents into a vector of term/token counts. It also allows for text data pre-processing before generating the vector representation. This functionality makes it a highly adaptable text feature representation module.

COSINE SIMILARITY




```
Cosine Similarity between A and B:0.7241379310344827
Cosine Distance between A and B:0.27586206896551735
```

Cosine Similarity is a measure of how similar two vectors in an inner product space are. The Cosine Similarity between two vectors, A and B, is calculated as

Cosine Similarity = (Ai2Bi2) / (AiBi)

This tutorial shows how to compute the Cosine Similarity between vectors in Python using NumPy functions.

IMPLEMENTATION

The ML approach for movie recommendation uses the Kaggle dataset. The structure of the movie dataset is shown in the below image. The dataset is taken from Dataset: https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata?select=tmdb_5000_movies.csv

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	budget	genres	homepage	id	keywords	original_la	original_ti	overview	popularity	productior	productior	release_da	revenue	runtime	spoken_la	status	tagline	title	vote_aver	vote_count
2	2.4E+08	[{"id": 28,	http://ww	19995	[{"id": 146	en	Avatar	In the 22n	150.438	[{"name":	[{"iso_316	########	2.8E+09	162	[{"iso_639	Released	Enter the	Avatar	7.2	11800
3	3E+08	[{"id": 12,	http://dis	285	[{"id": 270	en	Pirates of	Captain Ba	139.083	[{"name":	[{"iso_316	########	9.6E+08	169	[{"iso_639	Released	At the end	Pirates of	6.9	4500
4	2.5E+08	[{"id": 28,	http://ww	206647	[{"id": 470	en	Spectre	A cryptic r	107.377	[{"name":	[{"iso_316	########	8.8E+08	148	[{"iso_639	Released	A Plan No	Spectre	6.3	4466
5	2.5E+08	[{"id": 28,	http://ww	49026	[{"id": 849	en	The Dark K	Following	112.313	[{"name":	[{"iso_316	########	1.1E+09	165	[{"iso_639	Released	The Legen	The Dark K	7.6	9106
6	2.6E+08	[{"id": 28,	http://mo	49529	[{"id": 818	en	John Carte	John Carte	43.927	[{"name":	[{"iso_316	########	2.8E+08	132	[{"iso_639	Released	Lost in our	John Carte	6.1	2124
7	2.6E+08	[{"id": 14,	http://ww	559	[{"id": 851	en	Spider-Ma	The seemi	115.7	[{"name":	[{"iso_316	########	8.9E+08	139	[{"iso_639	Released	The battle	Spider-Ma	5.9	3576
8	2.6E+08	[{"id": 16,	http://dis	38757	[{"id": 156	en	Tangled	When the	48.682	[{"name":	[{"iso_316	########	5.9E+08	100	[{"iso_639	Released	They're tal	Tangled	7.4	3330
9	2.8E+08	[{"id": 28,	http://ma	99861	[{"id": 882	en	Avengers:	When Ton	134.279	[{"name":	[{"iso_316	########	1.4E+09	141	[{"iso_639	Released	A New Age	Avengers:	7.3	6767
10	2.5E+08	[{"id": 12,	http://har	767	[{"id": 616	en	Harry Pott	As Harry b	98.8856	[{"name":	[{"iso_316	########	9.3E+08	153	[{"iso_639	Released	Dark Secre	Harry Pott	7.4	5293
11	2.5E+08	[{"id": 28,	http://ww	209112	[{"id": 849	en	Batman v	Fearing th	155.79	[{"name":	[{"iso_316	########	8.7E+08	151	[{"iso_639	Released	Justice or	Batman v	5.7	7004
12	2.7E+08	[{"id": 12,	http://ww	1452	[{"id": 83,	en	Superman	Superman	57.9256	[{"name":	[{"iso_316	########	3.9E+08	154	[{"iso_639	Released		Superman	5.4	1400
13	2E+08	[{"id": 12,	http://ww	10764	[{"id": 627	en	Quantum	Quantum	107.929	[{"name":	[{"iso_316	########	5.9E+08	106	[{"iso_639	Released	For love, f	Quantum	6.1	2965
14	2E+08	[{"id": 12,	http://dis	58	[{"id": 616	en	Pirates of	Captain Ja	145.847	[{"name":	[{"iso_316	########	1.1E+09	151	[{"iso_639	Released	Jack is bac	Pirates of	7	5246
15	2.6E+08	[{"id": 28,	http://dis	57201	[{"id": 155	en	The Lone F	The Texas	49.047	[{"name":	[{"iso_316	########	8.9E+07	149	[{"iso_639	Released	Never Tak	The Lone F	5.9	2311
16	2.3E+08	[{"id": 28,	http://ww	49521	[{"id": 83,	en	Man of Ste	A young b	99.398	[{"name":	[{"iso_316	########	6.6E+08	143	[{"iso_639	Released	You will be	Man of Ste	6.5	6359
17	2.3E+08	[{"id": 12, "name": "A		2454	[{"id": 818	en	The Chron	One year a	53.9786	[{"name":	[{"iso_316	########	4.2E+08	150	[{"iso_639	Released	Hope has	The Chron	6.3	1630
18	2.2E+08	[{"id": 878	http://ma	24428	[{"id": 242	en	The Aveng	When an u	144.449	[{"name":	[{"iso_316	########	1.5E+09	143	[{"iso_639	Released	Some asse	The Aveng	7.4	11776
19	3.8E+08	[{"id": 12,	http://dis	1865	[{"id": 658	en	Pirates of	Captain Ja	135.414	[{"name":	[{"iso_316	########	1E+09	136	[{"iso_639	Released	Live Forev	Pirates of	6.4	4948
20	2.3E+08	[{"id": 28,	http://ww	41154	[{"id": 437	en	Men in Bla	Agents J (V	52.0352	[{"name":	[{"iso_316	########	6.2E+08	106	[{"iso_639	Released	They are b	Men in Bla	6.2	4160
21	2.5E+08	[{"id": 28,	http://ww	122917	[{"id": 417	en	The Hobbi	Immediate	120.966	[{"name":	[{"iso_316	########	9.6E+08	144	[{"iso_639	Released	Witness th	The Hobbi	7.1	4760
22	2.2E+08	[{"id": 28,	http://ww	1930	[{"id": 187	en	The Amazi	Peter Park	89.8663	[{"name":	[{"iso_316	########	7.5E+08	136	[{"iso_639	Released	The untold	The Amazi	6.5	6586
23	2E+08	[{"id": 28,	http://ww	20662	[{"id": 414	en	Robin Hoo	When sold	37.6683	[{"name":	[{"iso_316	########	3.1E+08	140	[{"iso_639	Released	Rise and ri	Robin Hoo	6.2	1398
24	2.5E+08	[{"id": 12,	http://ww	57158	[{"id": 603	en	The Hobbi	The Dwarv	94.3706	[{"name":	[{"iso_316	########	9.6E+08	161	[{"iso_639	Released	Beyond da	The Hobbi	7.6	4524
25	1.8E+08	[{"id": 12,	http://ww	2268	[{"id": 392	en	The Golde	After over	42.9909	[{"name":	[{"iso_316	########	3.7E+08	113	[{"iso_639	Released	There are	The Golde	5.8	1303
26	2.1E+08	[{"id": 12, "name": "A		254	[{"id": 774	en	King Kong	In 1933 Ne	61.226	[{"name":	[{"iso_316	########	5.5E+08	187	[{"iso_639	Released	The eighth	King Kong	6.6	2337
27	2E+08	[{"id": 18,	http://ww	597	[{"id": 258	en	Titanic	84 years la	100.026	[{"name":	[{"iso_316	########	1.8E+09	194	[{"iso_639	Released	Nothing o	Titanic	7.5	7562

movie_id	title	cast	crew
19995	Avatar	[{"cast_id"	[{"credit_id": "52fe48009251416c750aca23", "department": "Editing", "gender": 0, "id": 1721, "job": "Editor", "name": "Stephen E. Rivkin"}, {"credit_id": "539c47ecc3a36810e3001f87", "
285	Pirates of	[{"cast_id"	[{"credit_id": "52fe4232c3a36847f800b579", "department": "Camera", "gender": 2, "id": 120, "job": "Director of Photography", "name": "Dariusz Wolski"}, {"credit_id": "52fe4232c3a368
206647	Spectre	[{"cast_id"	[{"credit_id": "54805967c3a36829b5002c41", "department": "Sound", "gender": 2, "id": 153, "job": "Original Music Composer", "name": "Thomas Newman"}, {"credit_id": "52fe4d22c3a
49026	The Dark I	[{"cast_id"	[{"credit_id": "52fe4781c3a36847f81398c3", "department": "Sound", "gender": 2, "id": 947, "job": "Original Music Composer", "name": "Hans Zimmer"}, {"credit_id": "52fe4781c3a3684
49529	John Carte	[{"cast_id"	[{"credit_id": "52fe479ac3a36847f813eaa3", "department": "Writing", "gender": 2, "id": 7, "job": "Screenplay", "name": "Andrew Stanton"}, {"credit_id": "52fe479ac3a36847f813ea65", "
559	Spider-Ma	[{"cast_id"	[{"credit_id": "52fe4252c3a36847f80151a5", "department": "Production", "gender": 1, "id": 6410, "job": "Casting", "name": "Francine Maisler"}, {"credit_id": "52fe4252c3a36847f80151
38757	Tangled	[{"cast_id"	[{"credit_id": "52fe46db9251416c91062101", "department": "Production", "gender": 2, "id": 7879, "job": "Executive Producer", "name": "John Lasseter"}, {"credit_id": "52fe46db925141
99861	Avengers:	[{"cast_id"	[{"credit_id": "55d5f7d4c3a3683e7e0016eb", "department": "Sound", "gender": 2, "id": 531, "job": "Original Music Composer", "name": "Danny Elfman"}, {"credit_id": "55f7d488925141
767	Harry Pot	[{"cast_id"	[{"credit_id": "52fe4273c3a36847f801fab1", "department": "Camera", "gender": 0, "id": 2423, "job": "Director of Photography", "name": "Bruno Delbonnel"}, {"credit_id": "52fe4273c3a3
209112	Batman v	[{"cast_id"	[{"credit_id": "553bf23692514135c8002886", "department": "Sound", "gender": 2, "id": 947, "job": "Original Music Composer", "name": "Hans Zimmer"}, {"credit_id": "52fe4d5bc3a368
1452	Superman	[{"cast_id"	[{"credit_id": "553bef6a9251416874003c8f", "department": "Production", "gender": 2, "id": 3276, "job": "Casting", "name": "Roger Mussenden"}, {"credit_id": "553bee5592514135c800
10764	Quantum	[{"cast_id"	[{"credit_id": "52fe43b29251416c7501aa63", "department": "Writing", "gender": 2, "id": 455, "job": "Screenplay", "name": "Paul Haggis"}, {"credit_id": "52fe43b29251416c7501aab1", "
58	Pirates of	[{"cast_id"	[{"credit_id": "52fe4211c3a36847f8001873", "department": "Camera", "gender": 2, "id": 120, "job": "Director of Photography", "name": "Dariusz Wolski"}, {"credit_id": "52fe4211c3a368
57201	The Lone I	[{"cast_id"	[{"credit_id": "52fe4928c3a36847f818be95", "department": "Directing", "gender": 2, "id": 1704, "job": "Director", "name": "Gore Verbinski"}, {"credit_id": "52fe4928c3a36847f818bef9",
49521	Man of St	[{"cast_id"	[{"credit_id": "52fe4799c3a36847f813e667", "department": "Sound", "gender": 2, "id": 947, "job": "Original Music Composer", "name": "Hans Zimmer"}, {"credit_id": "52fe4799c3a3684
2454	The Chron	[{"cast_id"	[{"credit_id": "55a239e69251412979002e8b", "department": "Production", "gender": 1, "id": 1326, "job": "Casting", "name": "Liz Mullane"}, {"credit_id": "52fe4359c3a36847f804d729",
24428	The Aveng	[{"cast_id"	[{"credit_id": "52fe4495c3a368484e02b1cf", "department": "Sound", "gender": 2, "id": 37, "job": "Original Music Composer", "name": "Alan Silvestri"}, {"credit_id": "5496018d9251416e
1865	Pirates of	[{"cast_id"	[{"credit_id": "566b4f54c3a3683f56005151", "department": "Camera", "gender": 2, "id": 120, "job": "Director of Photography", "name": "Dariusz Wolski"}, {"credit_id": "52fe431cc3a368
41154	Men in Bl	[{"cast_id"	[{"credit_id": "52fe45b7c3a36847f80d68c7", "department": "Production", "gender": 2, "id": 488, "job": "Executive Producer", "name": "Steven Spielberg"}, {"credit_id": "52fe45b7c3a368
122917	The Hobbi	[{"cast_id"	[{"credit_id": "548ad49a9251414fa20011ab", "department": "Sound", "gender": 2, "id": 117, "job": "Original Music Composer", "name": "Howard Shore"}, {"credit_id": "553620f0c3a368
1930	The Amaz	[{"cast_id"	[{"credit_id": "5395a60dc3a368641d004492", "department": "Production", "gender": 1, "id": 6410, "job": "Casting", "name": "Francine Maisler"}, {"credit_id": "52fe4323c3a36847f803d
20662	Robin Hoo	[{"cast_id"	[{"credit_id": "52fe43f2c3a368484e0076ad", "department": "Production", "gender": 2, "id": 339, "job": "Producer", "name": "Brian Grazer"}, {"credit_id": "536a0d66c3a3681241007de0"
57158	The Hobbi	[{"cast_id"	[{"credit_id": "52fe4926c3a36847f818b787", "department": "Sound", "gender": 2, "id": 117, "job": "Original Music Composer", "name": "Howard Shore"}, {"credit_id": "52fe4926c3a368
2268	The Golde	[{"cast_id"	[{"credit_id": "52fe4348c3a36847f8048635", "department": "Art", "gender": 1, "id": 8384, "job": "Set Decoration", "name": "Anna Pinnock"}, {"credit_id": "52fe4348c3a36847f8048673"
254	King Kong	[{"cast_id"	[{"credit_id": "52fe422ec3a36847f800a1d7", "department": "Sound", "gender": 2, "id": 1213, "job": "Original Music Composer", "name": "James Newton Howard"}, {"credit_id": "52fe42
597	Titanic	[{"cast_id"	[{"credit_id": "52fe425ac3a36847f8017985", "department": "Production", "gender": 1, "id": 1262, "job": "Casting", "name": "Mali Finn"}, {"credit_id": "52fe425ac3a36847f8017967", "d
271110	Captain A	[{"cast_id"	[{"credit_id": "569443d59251414b67000428", "department": "Art", "gender": 2, "id": 2529, "job": "Set Decoration", "name": "Ronald R. Reiss"}, {"credit_id": "56e5521d92514115ec003f
44833	Battleship	[{"cast_id"	[{"credit_id": "52fe469ec3a36847f8108d45", "department": "Editing", "gender": 2, "id": 10816, "job": "Editor", "name": "Paul Rubell"}, {"credit_id": "53a158d00e0a266537002926", "de
135397	Jurassic W	[{"cast_id"	[{"credit_id": "52fe4bf7c3a368484e1a0683", "department": "Production", "gender": 2, "id": 488, "job": "Executive Producer", "name": "Steven Spielberg"}, {"credit_id": "554918b892514

The model development to recommend the movies using the vectorizer and cosine similarity implements the following steps:

Import NumPy, pandas. Download 2 datasets from Kaggle. Merge the datasets into 1 dataset. Read the movie dataset. Perform the data

cleaning operation and remove the unnamed columns. Change the new datatypes for the columns. Define a function to analyze the data. Remove the null values from the database. Create the attribute called tags by adding all the columns like overview, genres, keywords, cast , crew, etc

5 rows × 26 columns

```
In [12]: movies = movies[['movie_id_x','title','overview','genres','keywords','cast_x','crew_x']]
```

```
In [21]: movies.head()
```

Out[21]:

	movie_id_x	title	overview	genres	keywords	cast_x	
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", "...	"52fe480092
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...	"52fe4232c
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...	"54805967c3
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...	"52fe4781c
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...	"52fe479ac

```
In [28]: movies.isnull().sum()

Out[28]: movie_id_x    0
         title         0
         overview      0
         genres        0
         keywords      0
         cast_x        0
```

```
In [28]: movies.isnull().sum()

Out[28]: movie_id_x    0
         title         0
         overview      0
         genres        0
         keywords      0
         cast_x        0
         crew_x        0
         dtype: int64

In [31]: movies.dropna(inplace=True)

In [54]: movies.iloc[0].genres

Out[54]: ['Action', 'Adventure', 'Fantasy', 'Science Fiction']

In [81]: def convert(obj):
             L = []
             for i in ast.literal_eval(obj):
                 L.append(i['name'])
             return L

In [82]: movies['keywords'] = movies['keywords'].apply(convert)
```

Then we import nltk for removing all the duplicate words in the tags. You can use the Python library NLTK, or Natural Language Toolkit, for NLP. A large portion of the data that you might be examining is unstructured and contains text that can be read by humans. Preprocessing that data is necessary before you can programmatically evaluate it. After getting the final text then vectorization is applied to convert the text into vectors. An algorithm is vectorized when it is changed from dealing with a single value at a time to working with a set of values (vectors) at once. The process of vectorizing an algorithm involves changing it from using a single value at a time to using a set of values (vectors) at once. In the recommendation system, a machine learning algorithm is employed. A traditional method for transforming input data from its unprocessed form (text) into vectors of real numbers is called vectorization.

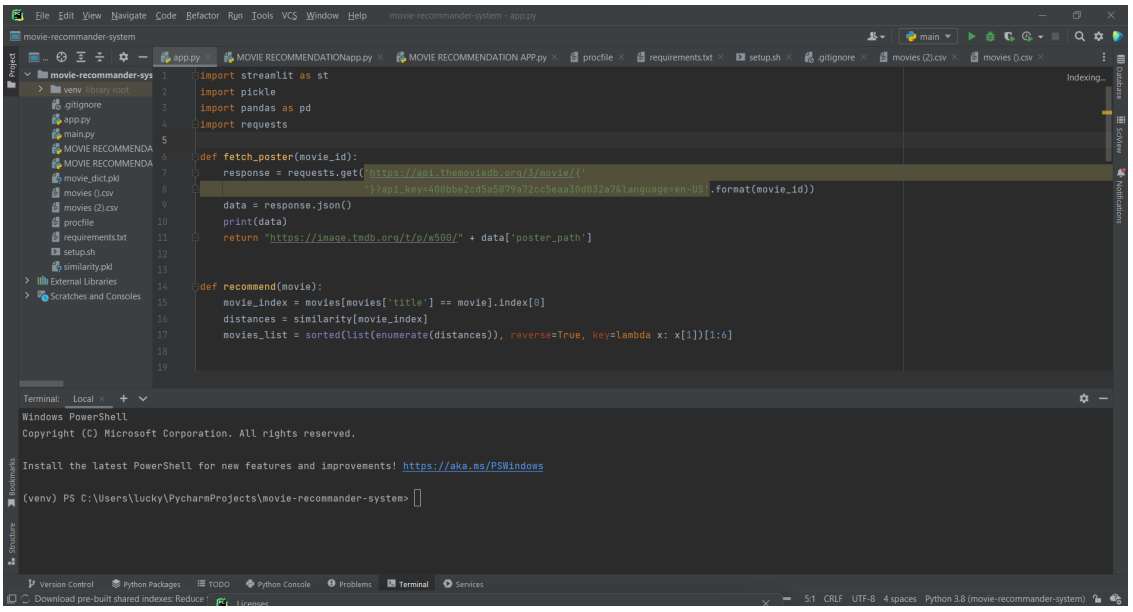
```
In [168]: from sklearn.metrics.pairwise import cosine_similarity

In [171]: similarity = cosine_similarity(vectors)

In [190]: similarity[0]

Out[190]: array([1.        , 0.08346223, 0.0860309 , ..., 0.04499213, 0.        ,
                 0.        ])
```

```
In [195]: def recommend(movie):
              movie_index = new_df[new_df['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(new_df.iloc[i[0]].title)

In [197]: recommend('Titanic')

          The Notebook
          Under the Same Moon
          Ghost Ship
          The Bounty
          Pirates of the Caribbean: On Stranger Tides
```

After the vectorization is done cosine similarity is used. cosine similarity finds the cosine distance between two vectors and finds the similarity score between two vectors. The similarity between two vectors is gauged using the cosine similarity metric. In particular, it ignores variations in the magnitude or scale of the vectors and

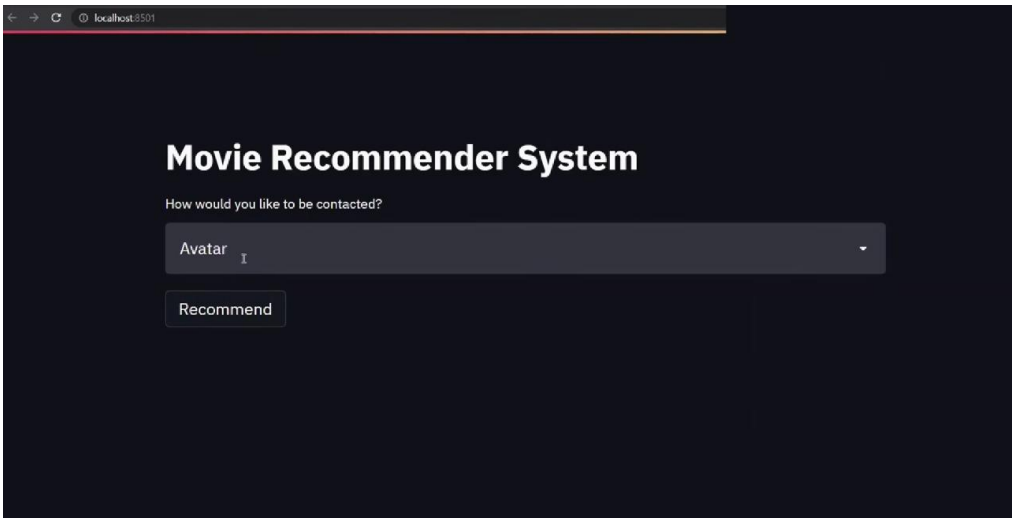
compares the similarity in their direction or orientation. To obtain a scalar using inner product multiplication, both vectors must belong to the same inner product space. The cosine of the angle between two vectors is used to determine how similar they are. so these give the similarity score between the movies. The main function is written after this cosine function. In the main function, the data includes thousands of films from different genres. One movie is entered into the recommendation system from the available data set, and five movies are returned as a recommendation result. Python programming is used to create a web-based platform utilizing the StreamLit platform.

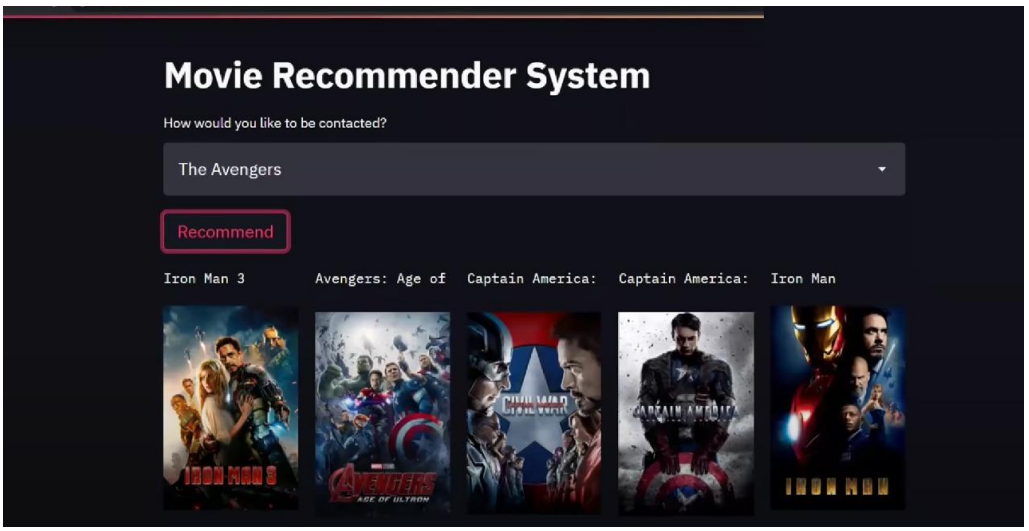
These systems, which are based on content recommendations, are considerate of people; they don't suggest anything to the user and let them make their own decisions.



RESULTS AND DISCUSSION

By implementing the above code,we can open a website as shown in the below image. The input column is visible where we give the input of a film name. We can see that For the recommendation system, we take input from the user first. For the output, we check the dedicated column which indicates the status(movie) of the data and show results depending on selected result. The recommendation system takes one movie as input from the given data set and outputs 5 movies as a recommendation result. In the tables here, we display movie recommendations. There recommendations of the movies are based on the similarity of the movie that is given as the input.




Movie Recommender System
How would you like to be contacted?
The Avengers
Recommend
Iron Man 3
Avengers: Age of
Captain America:
Captain America:
Iron Man

CONCLUSION AND FUTURE SCOPE

In this study, a count vectorizer and closest neighbor techniques are used to creating the movie recommendation model. The proposed recommendation model will eventually be expanded utilizing various soft computing approaches. The recommendation of movies using a content-based algorithm was the main emphasis of this work. This approach is based on key elements such as the film's quality, the directors, the entertainers, etc., which may also serve as inspiration. For upcoming work, a hybrid isolating methodology could be used to create the recommender plan rather than a content-based approach. Continuous evaluation reveals that cross-variety strategies are highly persuasive and provide more precise concepts. Consequently, a hybrid plan would be better.