Shri. Someshwar Shikshan Prasarak Mandal's

Sharadchandra Pawar College of Engineering and Technology

Someshwar Nagar, Tal. - Baramati, Dist. - Pune (412306)

Department of Computer Engineering

2022-23[6th Sem]

Mini-project Report of

Subject –

On

Movie Recommendation System.

Submitted by:

Roll Number	Name of Student	Seat Number
CO306	Bhosale Aishwarya Sanjay	T191194206
CO307	Bhandwalkar Rutik	T1911942307

Under the Guidance of Prof.

Pawar S. D.

DEPARTMENT OF COMPUTER ENGINEERING

Sharadchandra Pawar College of Engineering and Technology

SEMESTER – 6th 2022-23

CERTIFICATE

This is to certify that Project report entitled **"Movie Recommendation System"** is submitted in the partial fulfilment of requirement for the award of the Degree in Computer Engineering by Savitribai Phule Pune University as record of students' own work carried out by them under the guidance and supervision at Sharadchandra Pawar College Of Engineering And Technology Someshwar Nagar.

Name of Student	Seat Number
Bhosale Aishwarya Sanjay	T1911942306
Bhandwalkar Rutik	T1911942307

Place: Someshwar Nagar.

Date: / / 2023.

(Prof. Pawar S. D.) (Prof. Shah S. N.**)**

Guide ***Head of Computer Department***

ACKNOWLEDGEMENT

We have great pleasure and sense of satisfaction in presenting this mini-project report on "**Movie Recommendation System**" as part of the curriculum of Degree in Computer Engineering. Being novice in the field of designing and structuring in this mini-project on our own. We are very fortunate to be guided by people with vast and resourceful experience in their respective field of work.

We express our sincere gratitude to our guide "Prof. Pawar S. D. (Lecturer.)" for her timely guidance, support and suggestions. I am also thankful for her sincere help and for making us available all the facilities of the department. Without her efforts and constant monitoring of the mini-project and documentation, would not have been duly completed. Also, we express our sincere thanks to "Prof. Shah S. N.(HOD Computer Department)". Besides, we take this opportunity to express our sincere gratitude to the "Principal Dr. Sanjay Deokar." for providing a good environment and facilities to complete this mini-project. We would also like to thank all my colleagues who have directly or indirectly guided and helped us in the preparation of this mini-project.

Kawale Sagar.

Kambale Abhishek.

Part B: - Mini-project Proposal

Title: - ***Movie Recommendation System***

1.0 Problem Statement:-

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

2.0 Objective:-

The objective of this recommendation system is to provide satisfactory movie recommendations to users while keeping the system user friendly ie. by taking minimum input from users. It recommends the movies based on metadata of the movies and past user ratings.

3.0 Technology Used:-

Machine Learning Library:

- pandas ○ numpy
- Matplotlib
- difflib
- AST ○ scikit-learn.

4.0 Requirements:-

1) Python 3.6
2) Jupyter Notebook.

5.0 Theory:-

1) What is Scikit-Learn (Sklearn):-

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 consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

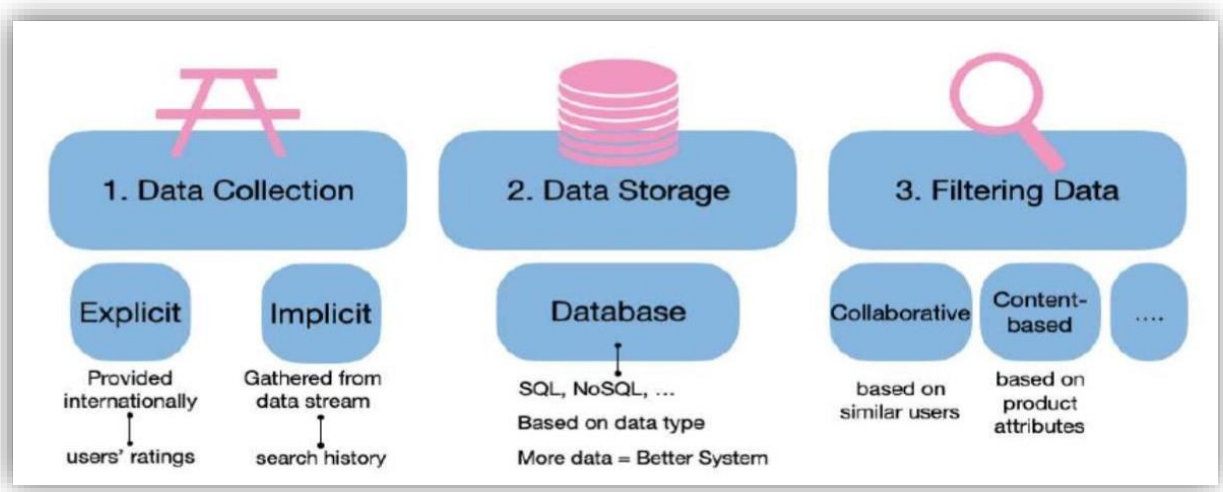
2) What is a Recommendation System?

Simply put a Recommendation System is a filtration program whose prime goal is to predict the “rating” or “preference” of a user towards a domain-specific item or item. In our case, this domain-specific item is a movie, therefore the main focus of recommendation

system is to filter and predict only those movies which a user would prefer given some data about the user him or herself.

3) *Recommendation System Mechanism*

The engine of the recommendation system filters the data via different machine learning algorithms, and based on that filtering, it can predict the most relevant entities to be recommended. After studying the previous behaviors of the users, it recommends products/services that the used may be interested on.



The engine's working of a recommendation is classified in these 3 steps:

- *Data Collection*

  The techniques that can be used to collect data are:

  1. Explicit, where data are provided intentionally as an information (e.g. user's input suchas movies rating)
  2. Implicit, where data are provided intentionally but gathered from available data stream(e.g. search history, clicks, order history, etc…)

- *Data Storage*

  It can be stored in a cloud storage such as SQL database, NoSQL database, or some other kind of object storage. However, it depends on the data type and amount as well. The more data that the storage can have for the model, the better recommendation system can be.

- *Filtering Data*

What is Collaborative Filtering?

Collaborative filtering is a technique that can filter out items that a user might like on the basis of reactions by similar users. It works by searching a large group of people and finding a smaller set of users with tastes similar to a particular user.

What is Content-based filtering?

Content-based filtering is a type of recommender system that attempts to guess what a user may like based on that user's activity. Content-based filtering makes recommendations by using keywords and attributes assigned to objects in a database (e.g., items in an online marketplace) and matching them to a user profile.

User-based Collaborative filtering:

The basic idea here is to find users that have similar past preference patterns as the user 'A' has had and then recommending him or her items liked by those similar users which 'A' has not encountered yet. This is achieved by making a matrix of items each user has rated/viewed/liked/clicked depending upon the task at hand, and then computing the similarity score between the users and finally recommending items that the concerned user isn't aware of but users similar to him/her are and liked it. For example, if the user 'A' likes 'Batman Begins', 'Justice League' and 'The Avengers' while the user 'B' likes 'Batman Begins', 'Justice League' and 'Thor' then they have similar interests because we know that these movies belong to the super-hero genre. So, there is a high probability that the user 'A' would like 'Thor' and the user 'B' would like The Avengers'.

Disadvantages:

1. People are fickle-minded i.e their taste change from time to time and as this algorithms based on user similarity it may pick up initial similarity patterns between 2 users who after a while may have completely different preferences
2. There are many more users than items therefore it becomes very difficult to maintain such large matrices and therefore needs to be recomputed very regularly.
3. This algorithm is very susceptible to shilling attacks where fake users profiles consisting of biased preference patterns are used to manipulate key decisions.

Item-based Collaborative Filtering:-

The concept in this case is to find similar movies instead of similar users and then recommending similar movies to that 'A' has had in his/her past preferences. This is executed by finding every pair of items that were rated/viewed/liked/clicked by the same user, then measuring the similarity of those rated/viewed/liked/clicked across all user who rated/viewed/liked/clicked both, and finally recommending them based on similarity scores. Here, for example, we take 2 movies 'A' and 'B' and check their ratings by all users who have rated both the movies and based on the similarity of these ratings, and based on this rating similarity by users who have rated both we find similar movies. So if most common users have rated 'A' and 'B' both similarly and it is highly probable that 'A' and 'B' are similar, therefore if someone has watched and liked 'A' they should be recommended 'B' and vice versa.

Advantages over User-based Collaborative Filtering : 1. Unlike people's taste, movies don't change.

2. There are usually a lot fewer items than people, therefore easier to maintain and computethe matrices.
3. Shilling attacks are much harder because items cannot be faked.

6.0 Code:-

```
import pandas as pd import
numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity df =
pd.read_csv("movie_dataset.csv") features =
['keywords','cast','genres','director'] def
combine_features(row):
return row['keywords'] +" "+row['cast']+" "+row["genres"]+" "+row["director"] for
feature in features:
df[feature] = df[feature].fillna('')
df["combined_features"] = df.apply(combine_features,axis=1) cv
= CountVectorizer()
count_matrix = cv.fit_transform(df["combined_features"])
cosine_sim = cosine_similarity(count_matrix) def
get_title_from_index(index):
return df[df.index == index]["title"].values[0]
def get_index_from_title(title): return
df[df.title == title]["index"].values[0]
movie_user_likes = "Avatar"
movie_index = get_index_from_title(movie_user_likes)
similar_movies =  list(enumerate(cosine_sim[movie_index]))
sorted_similar_movies = sorted(similar_movies,key=lambda x:x[1],reverse=True)[1:] i=0
print("Top 5 similar movies to "+movie_user_likes+" are:\n") for
element in sorted_similar_movies:
```

```
print(get_title_from_index(element[0])) i=i+1     if i>=5:
break
```

OUTPUT:-

Top 5 similar movies to Avatar are:

Guardians of the Galaxy

Aliens

Star Wars: Clone Wars: Volume 1

Star Trek Into Darkness

Star Trek Beyond

7.0 Conclusion:-

Recommendation systems have become an important part of everyone's lives. With the enormous number of movies releasing worldwide every year, people often miss out on some amazing work of arts due to the lack of correct suggestion. Putting machine learning based Recommendation systems into work is thus very important to get the right recommendations. We saw content-based recommendation systems that although may not seem very effective units own, but when combined with collaborative techniques can solve the cold start problems that collaborative filtering methods face when run independently.