



Data Science Project

- Project Name – Movie Recommendation System

- Submitting to:
 1. Mr. Mayur Dev Sewak
General Manager, Operations
Eisystems Services
 2. Ms. Mallika Srivastava
Trainer, Data Science & Analytics Domain
Eisystems Services

- 1. Submitting by:
Aum Subham Panda
Batch code – 23DAT002

Content Table

Serial No.	Title	Page No.
1	Cover Page	1
2	Content Table	2
3	List of Figures	2
4	Abstract of Project	3
5	Project Summary	3
6	Objectives of Project	4
7	Details of Project Developed	4-5
8	System Requirements	6
9	Input/Output Datasets (Screenshots)	6-8
10	Text code/Program	9-13
11	References	14

List of Figures

1. Details of Project Developed –
 - Caption: Details of Project developed
 - Page No: 5
 - Figure No:1

Abstract of Project

Project Title- Movie Recommendation System

1. A recommendation system is a system that provides suggestions to users for certain resources like books, movies, songs, etc., based on some data set.
2. Movie recommendation systems usually predict what movies a user will like based on the attributes present in previously liked movies.
3. Such recommendation systems are beneficial for organizations that collect data from large amounts of customers and wish to effectively provide the best suggestions possible.
4. A lot of factors can be considered while designing a movie recommendation system like the genre of the movie, the actors present in it or even the director of the movie. The systems can recommend movies based on one or a combination of two or more attributes.

Project Summary

Project Title- Movie Recommendation System

- The recommendation system analyzes the past preferences of the user concerned, and then it uses this information to try to find similar movies. This information is available in the database (e.g., lead actors, director, genre, etc.). After that, the system provides movie recommendations for the user.
- Movie recommendation systems use a set of different filtration strategies and algorithms to help users find the most relevant films. The most popular categories of ML algorithms used for movie recommendations include content-based filtering and collaborative filtering systems.
- In my project I have focussed on collaborative filtering systems.
- As the name suggests, this filtering strategy is based on the combination of the relevant user's and other users' behaviors. The system compares and contrasts these behaviors for the most optimal results. It's a collaboration of multiple users' film preferences and behaviors.
- The core element in this movie recommendation system and the ML algorithm it's built on is the history of all users in the database. Basically, collaborative filtering is based on the interaction of all users in the system with the items (movies). Thus, every user impacts the final outcome of this ML-based recommendation system, while content-based filtering depends strictly on the data from one user for its modelling.

Objectives of Project

Project Title- Movie Recommendation System

- The primary goal of movie recommendation systems is to filter and predict only those movies that a corresponding user is most likely to want to watch
- The ML algorithms for these recommendation systems use the data about this user from the system's database.
- This data is used to predict the future behavior of the user concerned based on the information from the past.
- Movie Recommendation System recommends the user the top 3 movies he/she can watch based on their previous movie ratings.
- Movie Recommendation System also tells the user which genre those top 3 recommended movies belong to.
- Movie Recommendation System will also show the top 5 movies of the genres of those top 3 movies based on various user ratings.

Details of Project Developed

Project Title- Movie Recommendation System

- This system is still in the code/program phase and has not been developed into software yet.
- The standard form of this system is "Read Data- Collect name of the user as input- Process the input- Generate necessary output in form of recommendations"
- There are 5 steps in this movie recommendation system project.
- Step 1- Extraction of data from CSV files into data frames.
- Step 2 - Storing the name of the user into a variable as that will be the key search element.
- Step 3- Recommendation Function which should give the top 3 movies recommended for a given user's name.
- Step 4 - Recommendation Function which should recommend the top 3 genres for each person.
- Step 5 - Function that will give the top 5 movies in each of those 3 genres for each person based on multiple user ratings.

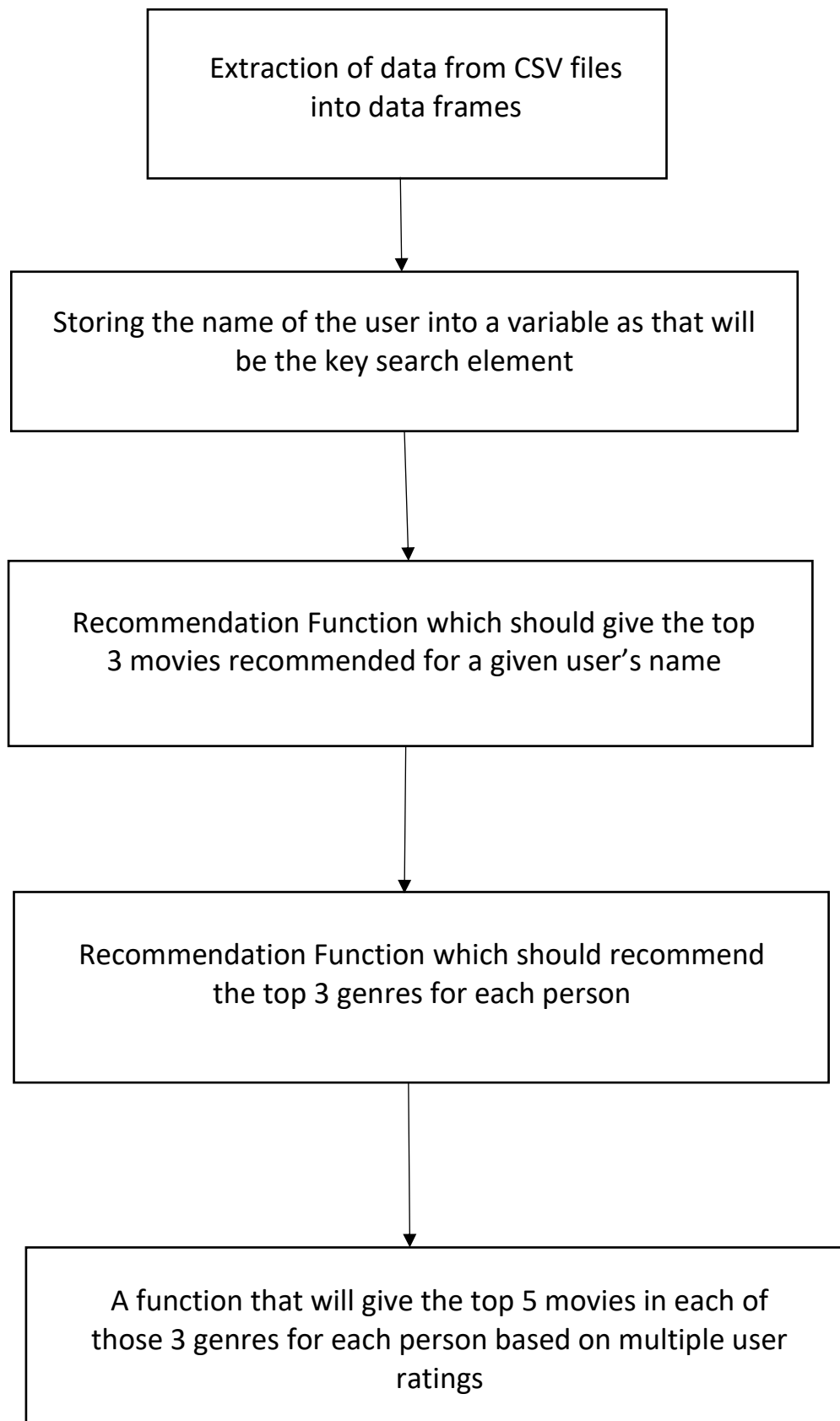


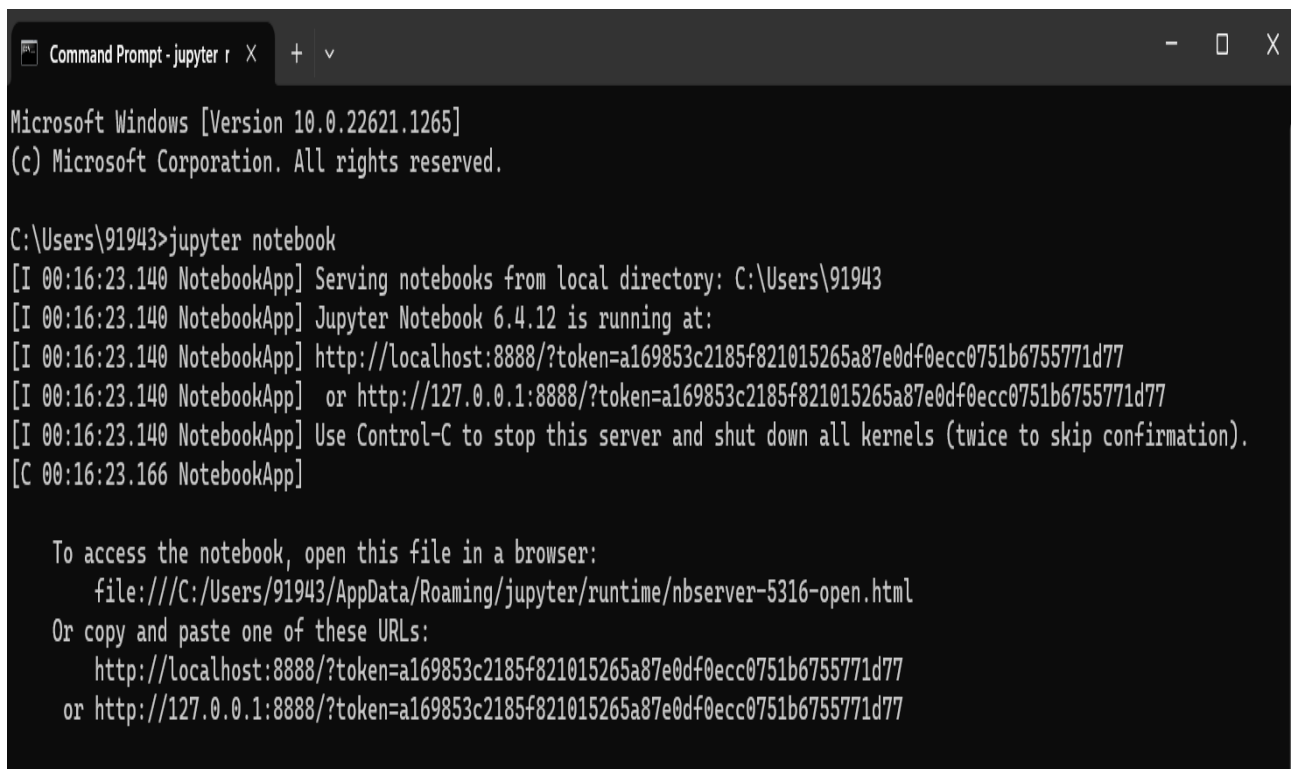
Fig. Details of Project Developed

System Requirements

- Windows 10 / Windows 11
- Python Version 3.7 or 3.8 or 3.9 or 4.0
- Jupyter Notebook
- Command Prompt

Input/Output Datasets (Screenshots)

1. Open Jupyter Notebook on Command Prompt

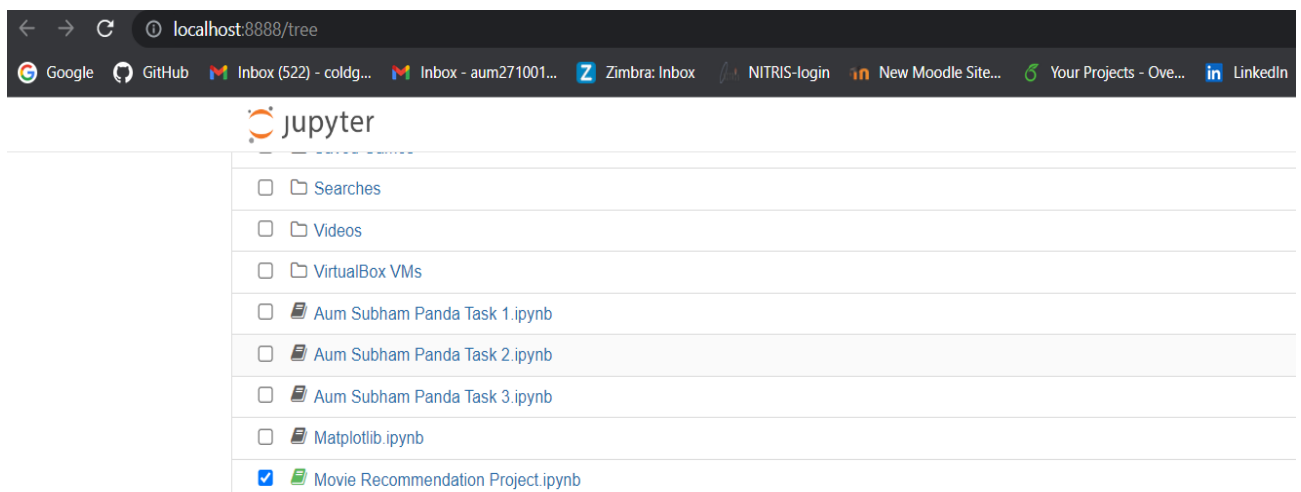


```
Microsoft Windows [Version 10.0.22621.1265]
(c) Microsoft Corporation. All rights reserved.

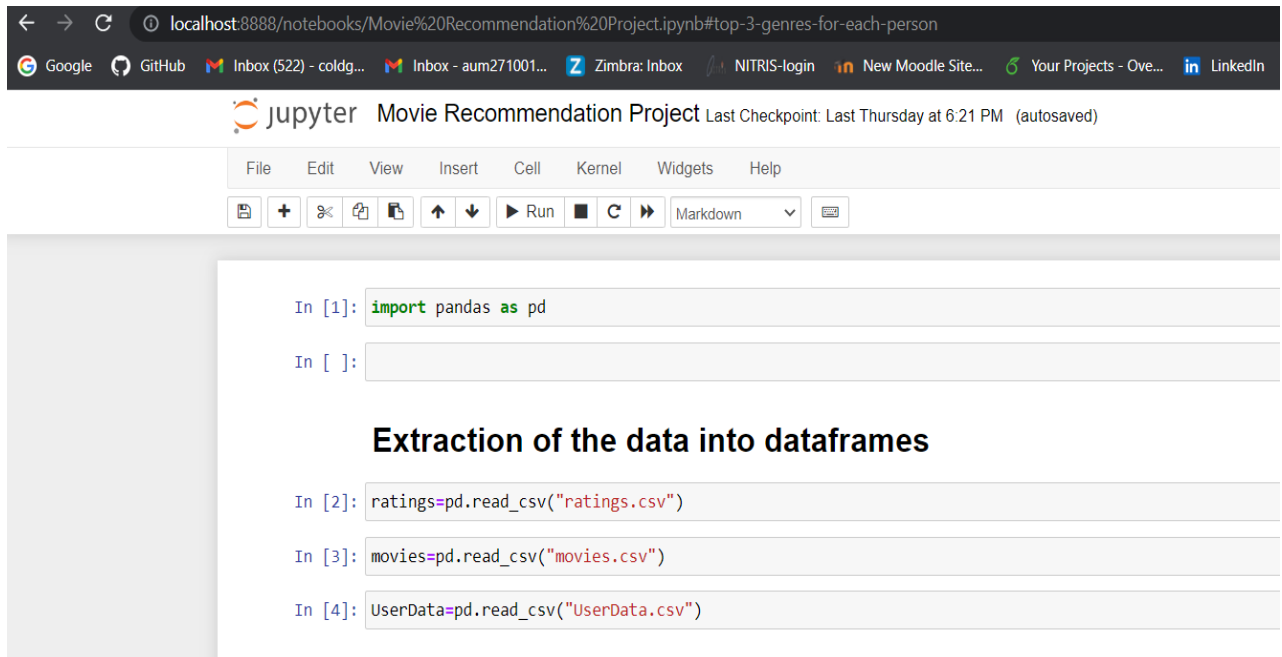
C:\Users\91943>jupyter notebook
[I 00:16:23.140 NotebookApp] Serving notebooks from local directory: C:\Users\91943
[I 00:16:23.140 NotebookApp] Jupyter Notebook 6.4.12 is running at:
[I 00:16:23.140 NotebookApp] http://localhost:8888/?token=a169853c2185f821015265a87e0df0ecc0751b6755771d77
[I 00:16:23.140 NotebookApp] or http://127.0.0.1:8888/?token=a169853c2185f821015265a87e0df0ecc0751b6755771d77
[I 00:16:23.140 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 00:16:23.166 NotebookApp]

To access the notebook, open this file in a browser:
    file:///C:/Users/91943/AppData/Roaming/jupyter/runtime/nbserver-5316-open.html
Or copy and paste one of these URLs:
    http://localhost:8888/?token=a169853c2185f821015265a87e0df0ecc0751b6755771d77
    or http://127.0.0.1:8888/?token=a169853c2185f821015265a87e0df0ecc0751b6755771d77
```

2. Accessing Movie Recommendation Project in local host



3. Movie Recommendation Project open and ready to use



```
In [1]: import pandas as pd

In [ ]:

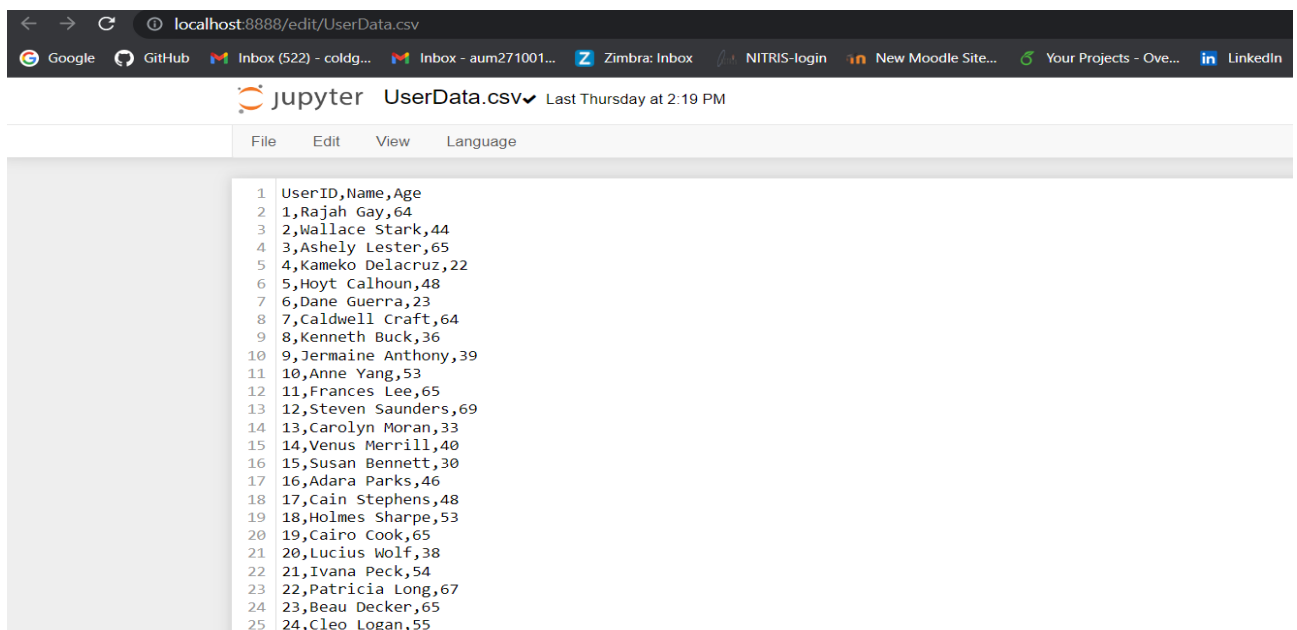
Extraction of the data into dataframes

In [2]: ratings=pd.read_csv("ratings.csv")

In [3]: movies=pd.read_csv("movies.csv")

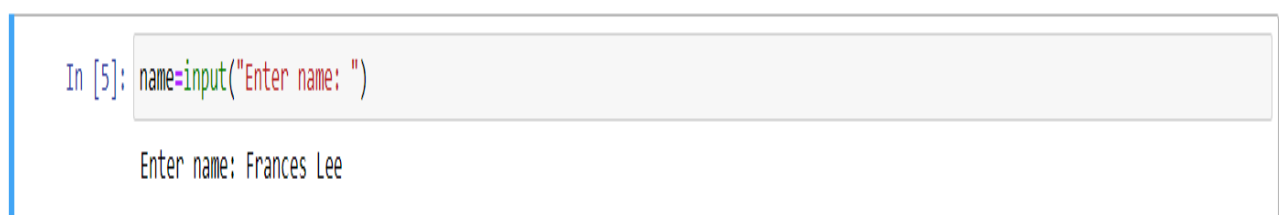
In [4]: UserData=pd.read_csv("UserData.csv")
```

4. Choosing one name of a user from 621 names in UserData.csv so as feed the same name as the search key element into the beginning of the program.



```
1 UserID,Name,Age
2 1,Rajah Gay,64
3 2,Wallace Stark,44
4 3,Ashely Lester,65
5 4,Kameko Delacruz,22
6 5,Hoyt Calhoun,48
7 6,Dane Guerra,23
8 7,Caldwell Craft,64
9 8,Kenneth Buck,36
10 9,Jermaine Anthony,39
11 10,Anne Yang,53
12 11,Frances Lee,65
13 12,Steven Saunders,69
14 13,Carolyn Moran,33
15 14,Venus Merrill,40
16 15,Susan Bennett,30
17 16,Adara Parks,46
18 17,Cain Stephens,48
19 18,Holmes Sharpe,53
20 19,Cairo Cook,65
21 20,Lucius Wolf,38
22 21,Ivana Peck,54
23 22,Patricia Long,67
24 23,Beau Decker,65
25 24,Cleo Logan,55
```

Storing name of user into a variable as that will be the key search element



```
In [5]: name=input("Enter name: ")

Enter name: Frances Lee
```

5. Recommendation Function which should give the top 3 movies recommended for a given user's name.

Recommendation Function, This should give the top 3 movies recommended for a user given name

Hunt for Red October, The (1990)

Payback (1999)

One Flew Over the Cuckoo's Nest (1975)

6. Recommendation Function which should recommend the top 3 genres for each person in accordance with the top 3 movies previously recommended.

top 3 genres for each person

['Action|Adventure|Thriller', 'Action|Thriller', 'Drama']

7. A function that will give the top 5 movies in each of those 3 genres for each person based on multiple user ratings.

top 5 movies in each of the top 3 genres

The corresponding top 5 movies of the top 3 genres are as follows :

Action|Adventure|Thriller ['Hunt for Red October, The (1990)', 'Rock, The (1996)', 'Casino Royale (2006)', 'GoldenEye (1995)', 'Goldfinger (1964)']

Action|Thriller ['In the Line of Fire (1993)', 'Enemy of the State (1998)', 'Payback (1999)', 'John Wick (2014)', 'Crank (2006)']

Drama ["One Flew Over the Cuckoo's Nest (1975)", 'Requiem for a Dream (2000)', 'Amadeus (1984)', 'Mr. Holland's Opus (1995)', 'Dead Poets Society (1989)']

Text code/Program

```
In [1]: import pandas as pd
```

```
In [ ]:
```

Extraction of the data into dataframes

```
In [2]: ratings=pd.read_csv("ratings.csv")
```

```
In [3]: movies=pd.read_csv("movies.csv")
```

```
In [4]: UserData=pd.read_csv("UserData.csv")
```

Storing name of user into a variable as that will be the key search element

```
In [5]: name=input("Enter name: ")
```

Enter name: Frances Lee

Recommendation Function, This should give the top 3 movies recommended for a user given name

```
In [6]: #recommendation function
def Recommend_func(name):
    for index,row in UserData.iterrows():
        if row['Name']==name:
            i=row['UserID']
            user_data=ratings.loc[ratings.userId==i]
            #important
            #list of all the movieIds watched by user
            l=list(user_data['movieId'])
            #important
            #getting all the movies which the user watched
            gen=[]
            for index,row in movies.iterrows():
                if row['movieId'] in l:
                    gen.append(row['genres'])
            #important
            #obtaining the highest watched genres by any user
            from collections import Counter
            count=Counter(gen)
            top=count.most_common(3)
            best_genres=[]
            for i in top:
                best_genres.append(i[0])
            #important
            best_movies=ratings.loc[ratings["rating"]==5]
            b_id=list(best_movies['movieId'])
```

```

#important
#finding all the movies of those genres
gen1=[]
gen2=[]
gen3=[]
for index,row in movies.iterrows():
    if row['genres']==best_genres[0]:
        gen1.append(row)
    elif row['genres']==best_genres[1]:
        gen2.append(row)
    elif row['genres']==best_genres[2]:
        gen3.append(row)
#important
#getting all the unwatched movies in each genre
unw1=[]
for i in gen1:
    if i['movieId'] not in l:
        unw1.append(i['movieId'])
unw2=[]
for i in gen2:
    if i['movieId'] not in l:
        unw2.append(i['movieId'])
unw3=[]
for i in gen3:
    if i['movieId'] not in l:
        unw3.append(i['movieId'])
#important
#Getting the count for all the highest rated movies
hm=Counter(b_id)
j=hm.most_common(5)

```

```

#getting the top 5 movie names of all time
top_5_id=[]
for i in j:
    top_5_id.append(i[0])
no_5stars=[]
for i in j:
    no_5stars.append(i[1])
movie_name=[]
for index,row in movies.iterrows():
    if row['movieId'] in top_5_id:
        movie_name.append(row['title'])
#important
#final movie recommended
best_of_gen1={}
for i in hm:
    if i in unw1:
        best_of_gen1[i]=hm[i]
t=list(best_of_gen1.values())
t.sort()
fc=t[-1]
for i in best_of_gen1:
    if best_of_gen1[i]==fc:
        gen1_final_mid=i

```

```

best_of_gen2={}
for i in hm:
    if i in unw2:
        best_of_gen2[i]=hm[i]
t=list(best_of_gen2.values())
t.sort()
fc=t[-1]
for i in best_of_gen2:
    if best_of_gen2[i]==fc:
        gen2_final_mid=i

best_of_gen3={}
for i in hm:
    if i in unw3:
        best_of_gen3[i]=hm[i]
t=list(best_of_gen3.values())
t.sort()
fc=t[-1]
for i in best_of_gen3:
    if best_of_gen3[i]==fc:
        gen3_final_mid=i
#important
#names of the final recommended movies
for i in gen1:
    if i['movieId']==gen1_final_mid:
        print(i['title'])
for i in gen2:
    if i['movieId']==gen2_final_mid:
        print(i['title'])
for i in gen3:
    if i['movieId']==gen3_final_mid:
        print(i['title'])

```

Recommend_func(name)

Hunt for Red October, The (1990)

Payback (1999)

One Flew Over the Cuckoo's Nest (1975)

top 3 genres for each person

```
In [7]: #top 3 genres
def Genres_func(name):
    for index,row in UserData.iterrows():
        if row['Name']==name:
            i=row['UserID']
            user_data=ratings.loc[ratings.userId==i]
            #important
            #list of all the movieIds watched by user
            l=list(user_data['movieId'])
            #important
            #getting all the movies which the user watched
            gen=[]
            for index,row in movies.iterrows():
                if row['movieId'] in l:
                    gen.append(row['genres'])
            #important
            #obtaining the highest watched genres by any user
            from collections import Counter
            count=Counter(gen)
            top=count.most_common(3)
            best_genres=[]
            for i in top:
                best_genres.append(i[0])
            return best_genres
l=Genres_func(name)
print(l)
```

```
['Action|Adventure|Thriller', 'Action|Thriller', 'Drama']
```

top 5 movies in each of the top 3 genres

```
In [8]: def top_5(genre):
        gen1=[]
        for index,row in movies.iterrows():
            if row['genres']==genre:
                gen1.append(row['movieId'])

        best_movies=ratings.loc[ratings["rating"]==5]
        b_id=list(best_movies['movieId'])
        b_in_g=[]
        for i in b_id:
            if i in gen1:
                b_in_g.append(i)
        from collections import Counter
        hm=Counter(b_in_g)
        j=hm.most_common(5)
        b=[]
        for i in j:
            b.append(i[0])
        final=[]
        for i in b:
            for index,row in movies.iterrows():
                if row['movieId']==i:
                    final.append(row['title'])
        return final

print("The corresponding top 5 movies of the top 3 genres are as follows : ")
print("")
for i in l:
    print(i,top_5(i))
```

The corresponding top 5 movies of the top 3 genres are as follows :

Action|Adventure|Thriller ['Hunt for Red October, The (1990)', 'Rock, The (1996)', 'Casino Royale (2006)', 'GoldenEye (1995)', 'Goldfinger (1964)']

Action|Thriller ['In the Line of Fire (1993)', 'Enemy of the State (1998)', 'Payback (1999)', 'John Wick (2014)', 'Crank (2006)']

Drama ['One Flew Over the Cuckoo's Nest (1975)', 'Requiem for a Dream (2000)', 'Amadeus (1984)', 'Mr. Holland's Opus (1995)', 'Dead Poets Society (1989)']

References

- <https://www.google.com/>
- <https://www.geeksforgeeks.org/>
- <https://labeledyourdata.com/articles/movie-recommendation-with-machine-learning#:~:text=The%20primary%20goal%20of%20movie,user%20from%20the%20system's%20database.>
- <https://www.sciencedirect.com/science/article/pii/S1110866516300470#:~:text=Movie%20recommendation%20systems%20provide%20a,websites%20and%20e%2Dcommerce%20applications.>
- <https://www.wikipedia.org/>