STC Jawwy

```
"""
Here we install libraries that are not installed by default
Example:  pyslsb
Feel free to add any library you are planning to use.
"""
!pip install pyxlsb
```

```
Collecting pyxlsb
  Downloading pyxlsb-1.0.10-py2.py3-none-any.whl (23 kB)
Installing collected packages: pyxlsb
Successfully installed pyxlsb-1.0.10
```

```
# Import the required libraries
"""
Please feel free to import any required libraries as per your needs
"""
import pandas as pd     # provides high-performance, easy to use structures and data analysis tools
import pyxlsb           # Excel extention to read xlsb files (the input file)
import numpy as np      # provides fast mathematical computation on arrays and matrices
from google.colab import drive
```

```
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Jawwy dataset

The dataset consists of details about each customer and the movies and/or tv shows watched in addition to the genre.

You are required to work on task three to build a recommendation engine for our platform to Recommend movies to usesrs that they might be interested in¶

```
dataframe = pd.read_excel("/content/drive/MyDrive/Colab Notebooks/STC Data Analysis/stc TV Data Set_T3.xlsb",index_col=0)
# Please make a copy of dataset if you are going to work directly and make changes on the dataset
# you can use   df=dataframe.copy()
```

```
# check the data shape
dataframe.shape
```

```
(1048575, 5)
```

```
# display the first 5 rows
dataframe.head()
```

	user_id_maped	program_name	rating	date_	program_genre
0	26138	100 treets	1	2017-05-27	Drama
1	7946	Moana	1	2017-05-21	Animation
2	7418	The Mermaid Princess	1	2017-08-10	Animation
3	19307	The Mermaid Princess	2	2017-07-26	Animation
4	15860	Churchill	2	2017-07-07	Biography

```
# describe the numeric values in the dataset
dataframe.describe()
```

	user_id_maped	rating
count	1.048575e+06	1.048575e+06
mean	1.709266e+04	2.497283e+00
std	1.003513e+04	1.119837e+00
min	1.000000e+00	1.000000e+00
25%	8.253000e+03	1.000000e+00
50%	1.714900e+04	2.000000e+00
75%	2.566500e+04	3.000000e+00
max	3.428000e+04	4.000000e+00

```
# check if any column has null value in the dataset
dataframe.isnull().any()
```

```
user_id_maped    False
program_name     False
rating           False
date_            False
program_genre    False
dtype: bool
```

```
# we import Visualization libraries
# you can ignore and use any other graphing libraries
import matplotlib.pyplot as plt # a comprehensive library for creating static, animated, and interactive visualizations
import plotly #a graphing library makes interactive, publication-quality graphs. Examples of how to make line plots, scatter plots,
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

```
"""
TODO build your Recommender system to Highlight Programs that usesrs might be interested in
"""
df = dataframe.copy()
```

```
df.head()
```

	user_id_maped	program_name	rating	date_	program_genre
0	26138	100 treets	1	2017-05-27	Drama
1	7946	Moana	1	2017-05-21	Animation
2	7418	The Mermaid Princess	1	2017-08-10	Animation
3	19307	The Mermaid Princess	2	2017-07-26	Animation
4	15860	Churchill	2	2017-07-07	Biography

```
movies_matrix = df.pivot_table(index=['program_name'], columns=['user_id_maped'], values='rating').fillna(0)
```

```
movies_matrix.head()
```

user_id_maped	1	5	9	11	15	17	20	26	28	30	...	34259	34261	34263	34265	34267	34269	34271	34273	34277	34
program_name																					
#FollowFriday	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	0.0	
10 Days in a Madhouse	0.0	0.0	0.0	0.0	1.5	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	
100 treets	0.0	0.0	0.0	1.0	2.0	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	
101 Dalmatians	0.0	0.0	0.0	0.0	2.0	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	

```
# Compress using scipy package
from scipy.sparse import csr_matrix

movies_matrix_sparse = csr_matrix(movies_matrix.values)


# Build the model
from sklearn.neighbors import NearestNeighbors
model = NearestNeighbors(metric='cosine', algorithm='brute')

# Fitting the model on our matrix
model.fit(movies_matrix_sparse)
```

```
NearestNeighbors(algorithm='brute', metric='cosine')
```

```
"""
TODO show the recommendations (top 5) for the people who watched "Moana" movie
"""
```

```
'\nTODO show the recommendations (top 5) for the people who watched "Moana" movie\n'
```

```
# Get the index of "Moana" movie
movies_m = movies_matrix.reset_index()
movies_m[['program_name']]
moana_index = movies_m.index[movies_m['program_name'] == "Moana"].tolist()[0]


# Calculate neighbour distances
n_recs = 5
distances, indices = model.kneighbors(movies_matrix.iloc[moana_index,:].values.reshape(1,-1), n_neighbors = n_recs+1)


# Get top five recommendations
topFive = []
for i in range(0, len(distances.flatten())):
    if i != 0:
      topFive.append({'Title':movies_matrix.index[indices.flatten()[i]], 'Distance':distances.flatten()[i]})
topFive = pd.DataFrame(topFive)
```

```
# Create a Styler object
styled_df = topFive.style

# Apply styles to the dataframe
#styled_df = styled_df.background_gradient(cmap='Blues')  # Apply a background gradient
#styled_df = styled_df.set_properties(**{'font-size': '12px'})  # Set font size
# Define custom CSS styles for zebra stripes
styles = [
    {'selector': 'thead th',
     'props': [('background-color', '#304543'), ('color', 'white'), ('text-align', 'center'), ('vertical-align', 'middle'), ('font-s
    {'selector': 'tbody tr:nth-child(even)',
     'props': [('background-color', '#dcccc0'), ('color', 'black')]},
    {'selector': 'tbody tr:nth-child(odd)',
     'props': [('background-color', '#f5f5ef'), ('color', 'black')]}
]

# Apply the custom styles
styled_df = styled_df.set_table_styles(styles)


styled_df = styled_df.set_properties(**{'font-size': '16px', 'text-align': 'center', 'vertical-align': 'middle'})  # Set font size a

print("Top 5 Recommendations for 'Moana' movie:")
topFive
```

```
Top 5 Recommendations for 'Moana' movie:
```

	Title	Distance
0	Trolls	0.427642
1	Surf's Up : WaveMania	0.470576
2	The Mermaid Princess	0.506638
3	The Boss Baby	0.551443
4	The Jetsons & WWE: Robo-WrestleMania!	0.561058

```
# Find the most genre rated
mean_ratings = df.groupby('program_genre')['rating'].mean()

print(mean_ratings)
```

```
program_genre
Action                              2.358474
Adventure                           2.125523
Animation                           2.613295
Biography                           1.821320
Comedy                              2.418921
Crime                               2.284653
Documentary                         2.011512
Drama                               2.643298
Family                              2.677141
Horror                              2.340451
NOT_DEFINED_IN_UMS                  1.886439
Romance                             2.935547
SERIES_NOT_ADDED_UNDER_ANY_GENRE    2.481481
Sci-Fi                              2.885779
Thriller                            2.316255
Wrestling                           1.125000
Name: rating, dtype: float64
```