

ASSIGNMENT 8

RECOMMENDER SYSTEM

NAHUSHA ACHARYA PES1201700044

KEERTHAN G PES1201700963

PUSHPENDER SINGH PES1201700243

- **Collaborative filtering:** Collaborative filtering approaches build a model from user's past behavior (i.e. items purchased or searched by the user) as well as similar decisions made by other users. This model is then used to predict items (or ratings for items) that user may have an interest in.
- **Content-based filtering:** Content-based filtering approaches uses a series of discrete characteristics of an item in order to recommend additional items with similar properties. Content-based filtering methods are totally based on a description of the item and a profile of the user's preferences. It recommends items based on user's past preferences.

Check out all the movies and their respective IDs

	user_id	item_id	rating	timestamp	title
0	0	50	5	881250949	Star Wars (1977)
1	290	50	5	880473582	Star Wars (1977)
2	79	50	4	891271545	Star Wars (1977)
3	2	50	5	888552084	Star Wars (1977)
4	8	50	5	879362124	Star Wars (1977)

Calculate mean rating of all movies

```
title
Marlene Dietrich: Shadow and Light (1996)    5.0
Prefontaine (1997)                           5.0
Santa with Muscles (1996)                   5.0
Star Kid (1997)                             5.0
Someone Else's America (1995)               5.0
Name: rating, dtype: float64
```

creating dataframe with 'rating' count values

collaborative filtering Last Checkpoint: a day ago (autosaved)

```

ratings = pd.DataFrame(data.groupby('title')['rating'].mean())
ratings['num of ratings'] = pd.DataFrame(data.groupby('title')['rating'].count())
ratings.head()

```

Out[7]:

	rating	num of ratings
title		
'Til There Was You (1997)	2.333333	9
1-900 (1994)	2.600000	5
101 Dalmatians (1996)	2.908257	109
12 Angry Men (1957)	4.344000	125
187 (1997)	3.024390	41

```

In [8]: # creating dataframe with 'rating' count values
ratings = pd.DataFrame(data.groupby('title')['rating'].mean())
ratings['num of ratings'] = pd.DataFrame(data.groupby('title')['rating'].count())

```

collaborative filtering Last Checkpoint: a day ago (autosaved)

Out[12]:

	rating	num of ratings
title		
Star Wars (1977)	4.359589	584
Contact (1997)	3.803536	509
Fargo (1996)	4.155512	508
Return of the Jedi (1983)	4.007890	507
Liar Liar (1997)	3.156701	485
English Patient, The (1996)	3.656965	481
Scream (1996)	3.441423	478
Toy Story (1995)	3.878319	452
Air Force One (1997)	3.631090	431
Independence Day (ID4) (1996)	3.438228	429

```

In [13]: # analysing correlation with similar movies
starwars_user_ratings = moviemat['Star Wars (1977)']

```

analysing correlation with similar movies

```
starwars_user_ratings = moviemat['Star Wars (1977)']
```

```
liarliar_user_ratings = moviemat['Liar Liar (1997)']
```

```
starwars_user_ratings.head()
```

```
user_id
0      5.0
1      5.0
2      5.0
3      NaN
4      5.0
Name: Star Wars (1977), dtype: float64
```

In [14]:

```
# analysing correlation with similar movies
```

```
similar_to_starwars = moviemat.corrwith(starwars_user_ratings)
```

```
similar_to_liarliar = moviemat.corrwith(liarliar_user_ratings)
```

```
corr_starwars = pd.DataFrame(similar_to_starwars, columns=['Correlation'])
```

```
corr_starwars.dropna(inplace = True)
```

```
corr_starwars.head()
```

The screenshot shows a Jupyter Notebook titled 'collaborative filtering' running on a local host. A red warning box at the top indicates a 'RuntimeWarning: divide by zero encountered in true_divide' from numpy. Below the warning, the output of a cell (Out[14]) is displayed as a table of movie correlations. The table has two columns: 'title' and 'Correlation'. The movies listed are 'Til There Was You (1997)', '1-900 (1994)', '101 Dalmatians (1996)', '12 Angry Men (1957)', and '187 (1997)'. The correlation values are 0.872872, -0.645497, 0.211132, 0.184289, and 0.027398 respectively. Below the table, the code for the next cell (In [15]) is visible, which sorts movies by correlation and joins the ratings data.

title	Correlation
'Til There Was You (1997)	0.872872
1-900 (1994)	-0.645497
101 Dalmatians (1996)	0.211132
12 Angry Men (1957)	0.184289
187 (1997)	0.027398

```

In [15]: # Similar movies like starwars
corr_starwars.sort_values('Correlation', ascending = False).head(10)
corr_starwars = corr_starwars.join(ratings['num of ratings'])
corr_starwars.head()

```

Similar movies as of liarliar

```
corr_liarliar = pd.DataFrame(similar_to_liarliar, columns=['Correlation'])
```

```
corr_liarliar.dropna(inplace = True)
```

```
corr_liarliar = corr_liarliar.join(ratings['num of ratings'])
```

```
corr_liarliar[corr_liarliar['num of ratings']>100].sort_values('Correlation', ascending = False).head()
```

collaborative filtering - Jupyter Notebook

localhost:8888/notebooks/Desktop/Data-Analytics-master/Data-Analytics-master/collaborative%20filtering.ipynb

jupyter collaborative filtering Last Checkpoint: a day ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Out[16]:

	Correlation	num of ratings
Liar Liar (1997)	1.000000	485
Batman Forever (1995)	0.516968	114
Mask, The (1994)	0.484650	129
Down Periscope (1996)	0.472681	101
Con Air (1997)	0.469828	137

In []:

12:40 PM 11/19/2019