movie-recommender-system

February 16, 2024

1 Python | Implementation of Movie Recommender System

Recommender System is a system that seeks to predict or filter preferences according to the user's choices. Recommender systems are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general.

1.0.1 Import Libraries and data

[152]:
```
import pandas as pd
```

[153]:
```
## Get the Data
column_names = ['user_id', 'item_id','rating','timestamp']
```

[154]:
```
path ="drive/MyDrive/Project 15 Regression/Data/file.tsv"
```

[155]:
```
df = pd.read_csv(path, sep = '\t', names = column_names)
```

[156]:
```
df.head()
```

[156]:
```
   user_id  item_id  rating  timestamp
0        0       50       5  881250949
1        0      172       5  881250949
2        0      133       1  881250949
3      196      242       3  881250949
4      186      302       3  891717742
```

[157]:
```
# Check out all the movies and their respective IDs
movie_titles = pd.read_csv("drive/MyDrive/Project 15 Regression/Data/
 ↪Movie_Id_Titles.csv")
movie_titles.head()
```

[157]:
```
   item_id              title
0        1   Toy Story (1995)
1        2   GoldenEye (1995)
2        3  Four Rooms (1995)
3        4  Get Shorty (1995)
4        5     Copycat (1995)
```

1.0.2 marging the two dataframes

[158]:

```
data = pd.merge(df, movie_titles, on = 'item_id')
data.head()
```

[158]:

```
   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)
```

[159]:

```
# Calculating Mean Rating OF ALL MOVIES
data.groupby('title')['rating'].mean().sort_values(ascending =False).head()
```

[159]:

```
title
They Made Me a Criminal (1939)                5.0
Marlene Dietrich: Shadow and Light (1996)     5.0
Saint of Fort Washington, The (1993)          5.0
Someone Else's America (1995)                 5.0
Star Kid (1997)                               5.0
Name: rating, dtype: float64
```

[160]:

```
# Calculate count rating of all movies
data.groupby('title')['rating'].count().sort_values(ascending=False).head()
```

[160]:

```
title
Star Wars (1977)             584
Contact (1997)               509
Fargo (1996)                 508
Return of the Jedi (1983)    507
Liar Liar (1997)             485
Name: rating, dtype: int64
```

1.0.3 Dataframe with rating count values

[161]:

```
# Dataframe with rating count values
ratings = pd.DataFrame(data.groupby('title')['rating'].mean())

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

[161]:

```
                             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
```

1.1 Visualization Imports

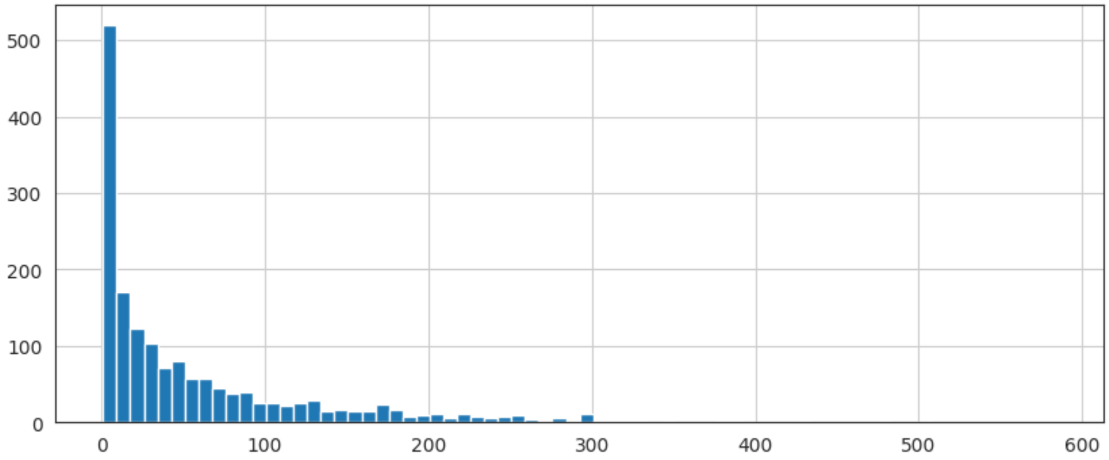
[162]:

```
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style('white')
%matplotlib inline
```

[163]:

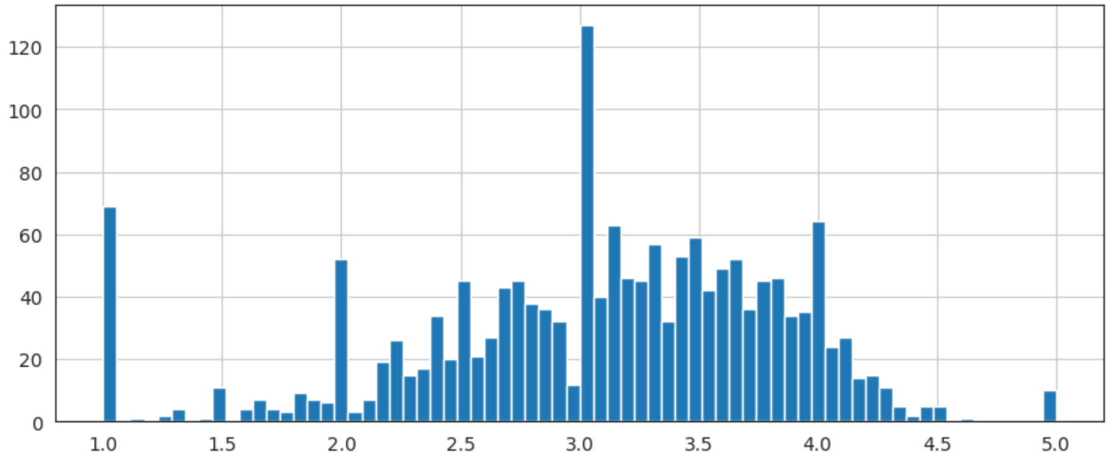
```
# Plot graph of num of ratings column
plt.figure(figsize=(10,4))
ratings['num of ratings'].hist(bins = 70)
plt.show()
```



[164]:

```
# plot graph of 'ratings' column
plt.figure(figsize =(10, 4))

ratings['rating'].hist(bins = 70)
plt.show()
```



Because the data is normaly disributed i will fill all the null values with the mean

```
[166]: print(data.isnull().sum())
```

```
user_id      0
item_id      0
rating       0
timestamp    0
title        0
dtype: int64
```

```
[174]: # Sorting Values according to the 'num of rating column'
moviemat = data.pivot_table(index='user_id', columns='title', values='rating')

# Sort 'ratings' dataframe and display the first 10 rows in one line
top_ratings = ratings.sort_values('num of ratings', ascending=False).head(10)
top_ratings
```

```
[174]:                                  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
```

[175]:

```
# Analysing Correlation with Similar movies
starwars_user_ratings = moviemat['Star Wars (1977)']
liarliar_user_ratings = moviemat['Liar Liar (1997)']

print(starwars_user_ratings.head(),liarliar_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 user_id
0    NaN
1    NaN
2    1.0
3    2.0
4    5.0
Name: Liar Liar (1997), dtype: float64
```

[181]:

```
# 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()
```

```
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:2889:
RuntimeWarning: Degrees of freedom <= 0 for slice
  c = cov(x, y, rowvar, dtype=dtype)
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:2748:
RuntimeWarning: divide by zero encountered in divide
  c *= np.true_divide(1, fact)
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:2889:
RuntimeWarning: Degrees of freedom <= 0 for slice
  c = cov(x, y, rowvar, dtype=dtype)
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:2748:
RuntimeWarning: divide by zero encountered in divide
  c *= np.true_divide(1, fact)
```

[181]:

```
                           Correlation
title
'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
```

[182]:

```
# Similar movies like starwars
corr_starwars.sort_values('Correlation', ascending = False)
corr_starwars = corr_starwars.join(ratings['num of ratings'])

corr_starwars.head()

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

[182]:

```
                                                    Correlation  \
title
Star Wars (1977)                                       1.000000
Empire Strikes Back, The (1980)                        0.748353
Return of the Jedi (1983)                              0.672556
Raiders of the Lost Ark (1981)                         0.536117
Austin Powers: International Man of Mystery (1997)     0.377433

                                                    num of ratings
title
Star Wars (1977)                                               584
Empire Strikes Back, The (1980)                                368
Return of the Jedi (1983)                                      507
Raiders of the Lost Ark (1981)                                 420
Austin Powers: International Man of Mystery (1997)             130
```

[183]:

```
# 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()
```

[183]:

```
                       Correlation  num of ratings
title
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
```