recommendation-system

November 25, 2024

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[2]:
```
data=pd.read_csv(r"/content/anime.csv")
```

[3]:
```
data
```

[3]:
```
       anime_id                                               name  \
0         32281                                     Kimi no Na wa.
1          5114                   Fullmetal Alchemist: Brotherhood
2         28977                                           Gintama°
3          9253                                        Steins;Gate
4          9969                                      Gintama&#039;
…         …                                                …
12289      9316       Toushindai My Lover: Minami tai Mecha-Minami
12290      5543                                        Under World
12291      5621                     Violence Gekiga David no Hoshi
12292      6133  Violence Gekiga Shin David no Hoshi: Inma Dens…
12293     26081                   Yasuji no Pornorama: Yacchimae!!

                                                   genre   type episodes  \
0                   Drama, Romance, School, Supernatural  Movie        1
1      Action, Adventure, Drama, Fantasy, Magic, Mili…     TV       64
2      Action, Comedy, Historical, Parody, Samurai, S…     TV       51
3                                       Sci-Fi, Thriller     TV       24
4      Action, Comedy, Historical, Parody, Samurai, S…     TV       51
…                                                  …    …      …
12289                                             Hentai    OVA        1
12290                                             Hentai    OVA        1
12291                                             Hentai    OVA        4
12292                                             Hentai    OVA        1
12293                                             Hentai  Movie        1

       rating  members
0        9.37   200630
```

```
1        9.26   793665
2        9.25   114262
3        9.17   673572
4        9.16   151266
…       …      …
12289    4.15      211
12290    4.28      183
12291    4.88      219
12292    4.98      175
12293    5.46      142

[12294 rows x 7 columns]
```

[4]:
```
data.shape
```

[4]:
```
(12294, 7)
```

[5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12294 entries, 0 to 12293
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   anime_id  12294 non-null  int64
 1   name      12294 non-null  object
 2   genre     12232 non-null  object
 3   type      12269 non-null  object
 4   episodes  12294 non-null  object
 5   rating    12064 non-null  float64
 6   members   12294 non-null  int64
dtypes: float64(1), int64(2), object(4)
memory usage: 672.5+ KB
```

[6]:
```
data.isnull().sum()
```

[6]:
```
anime_id      0
name          0
genre        62
type         25
episodes      0
rating      230
members       0
dtype: int64
```

[7]:
```
data.dropna(inplace=True)
```

[8]:

```
data.isnull().sum()
```

[8]:

```
anime_id    0
name        0
genre       0
type        0
episodes    0
rating      0
members     0
dtype: int64
```

[9]:

```
data.duplicated().sum()
```

[9]:

```
0
```

[10]:

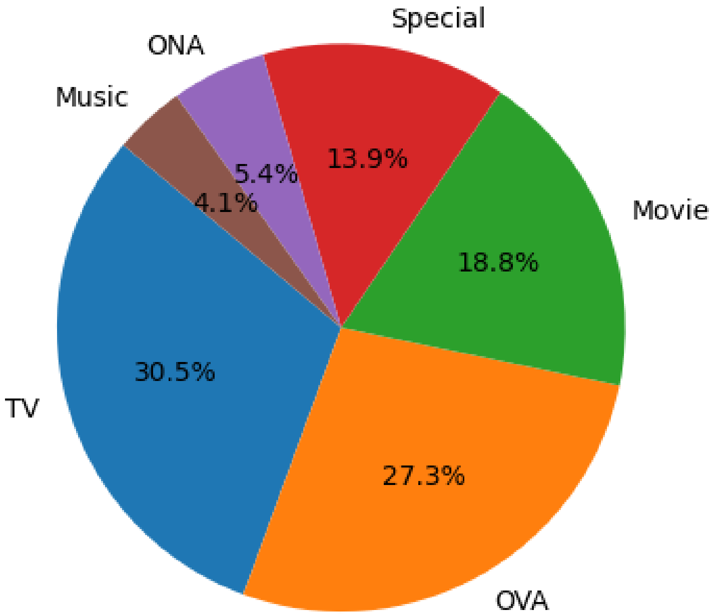
```
data.isnull().sum()
```

[10]:

```
anime_id    0
name        0
genre       0
type        0
episodes    0
rating      0
members     0
dtype: int64
```

[11]:

```
b=data['type'].value_counts()
b

labels = ['TV', 'OVA', 'Movie', 'Special','ONA','Music']
plt.pie(b, labels=labels, autopct='%1.1f%%', startangle=140)

### checking for different genre

data['genre'] = data['genre'].apply(lambda x: x.split(', '))
```

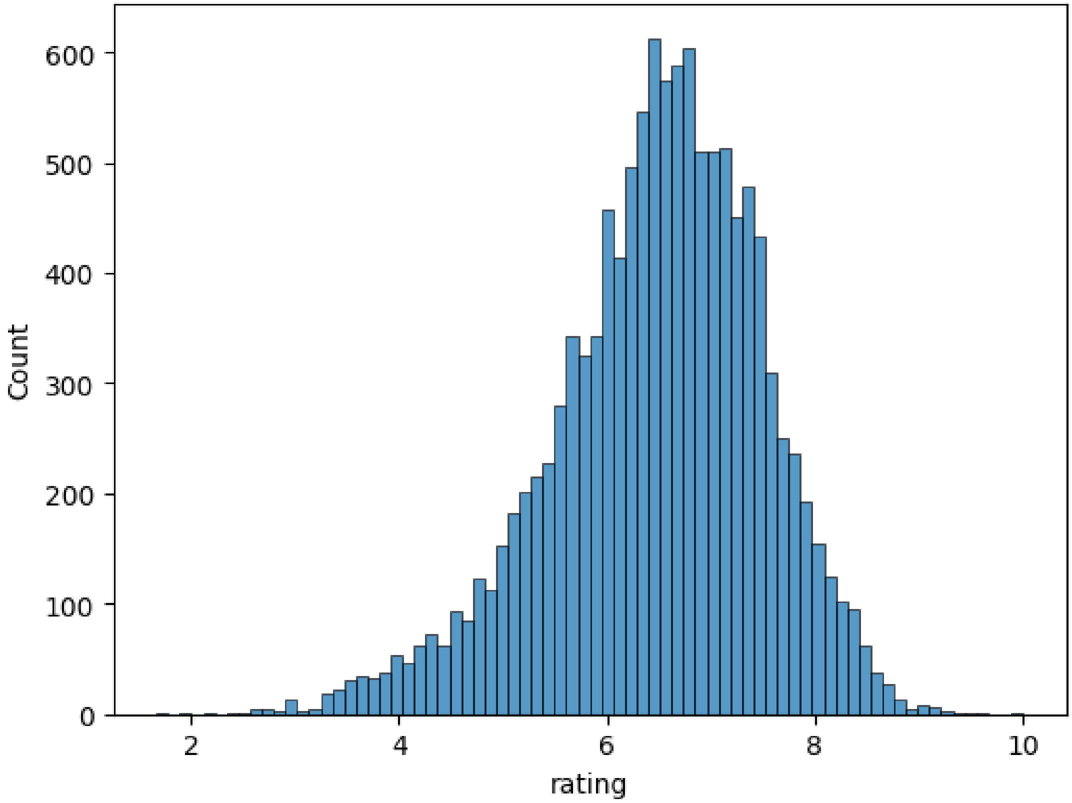


[12]:

```
data

data['rating'].info()

### lets check the distribution of the rating

sns.histplot(data['rating'])

### lets check the skewness of the plot

data['rating'].skew()

### the data is moderately distributed
```

```
<class 'pandas.core.series.Series'>
Index: 12017 entries, 0 to 12293
Series name: rating
Non-Null Count  Dtype
--------------  -----
12017 non-null  float64
dtypes: float64(1)
memory usage: 187.8 KB
```

```
[12]: -0.5443140848094782
```

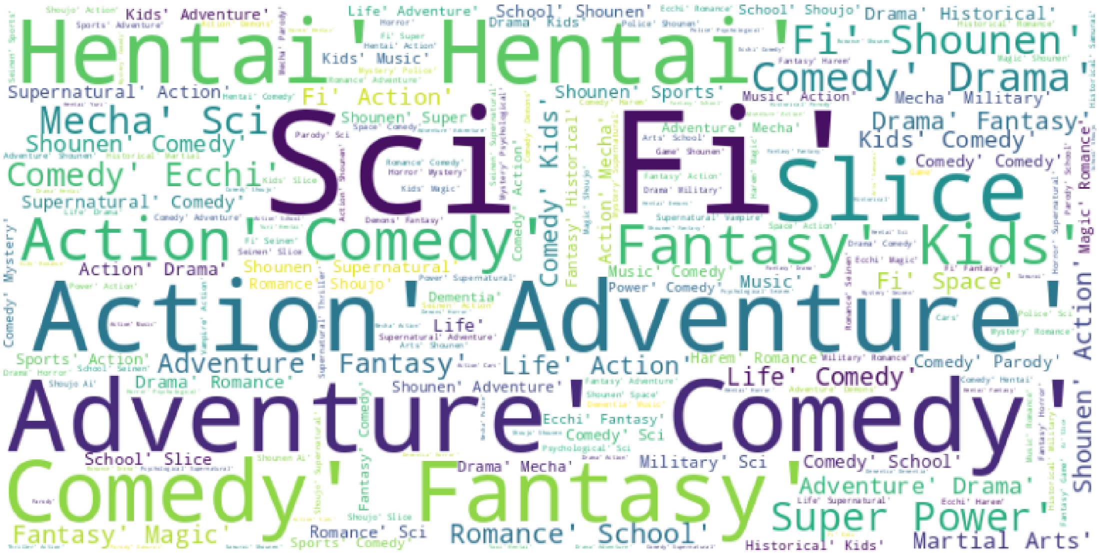


```
[13]: data['genre']
      exploded_series = data['genre'].explode()
      unique_elements = exploded_series.unique()

      print(unique_elements)
```

```
['Drama' 'Romance' 'School' 'Supernatural' 'Action' 'Adventure' 'Fantasy'
 'Magic' 'Military' 'Shounen' 'Comedy' 'Historical' 'Parody' 'Samurai'
 'Sci-Fi' 'Thriller' 'Sports' 'Super Power' 'Space' 'Slice of Life'
 'Mecha' 'Music' 'Mystery' 'Seinen' 'Martial Arts' 'Vampire' 'Shoujo'
 'Horror' 'Police' 'Psychological' 'Demons' 'Ecchi' 'Josei' 'Shounen Ai'
 'Game' 'Dementia' 'Harem' 'Cars' 'Kids' 'Shoujo Ai' 'Hentai' 'Yaoi'
 'Yuri']
```

```
[16]: import pandas as pd
      from wordcloud import WordCloud
      import matplotlib.pyplot as plt
```

```
# Convert NaN values to empty strings
data['genre'] = data['genre'].fillna('')

# Concatenate all genres into a single string
all_genres = ' '.join(data['genre'].astype(str))

# Generate a word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').
 ↪generate(all_genres)

# Display the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



```
[18]: import pandas as pd
      from sklearn.metrics.pairwise import cosine_similarity

      # Replace with the actual path to your anime.csv file
      anime_data = pd.read_csv("anime.csv")  # If the file is in the same directory␣
       ↪as the script
      # or
      # anime_data = pd.read_csv("/home/user/data/anime.csv") # If the file is in a␣
       ↪different directory, provide the full path

      anime_data['genre'] = anime_data['genre'].fillna('')
      anime_data
```

[18]:

```
       anime_id                                               name  \
0         32281                                     Kimi no Na wa.
1          5114                   Fullmetal Alchemist: Brotherhood
2         28977                                           Gintama°
3          9253                                        Steins;Gate
4          9969                                      Gintama&#039;
…         …                                                 …
12289      9316       Toushindai My Lover: Minami tai Mecha-Minami
12290      5543                                        Under World
12291      5621                     Violence Gekiga David no Hoshi
12292      6133  Violence Gekiga Shin David no Hoshi: Inma Dens…
12293     26081                   Yasuji no Pornorama: Yacchimae!!

                                                   genre   type episodes  \
0                   Drama, Romance, School, Supernatural  Movie        1
1      Action, Adventure, Drama, Fantasy, Magic, Mili…     TV       64
2      Action, Comedy, Historical, Parody, Samurai, S…     TV       51
3                                       Sci-Fi, Thriller     TV       24
4      Action, Comedy, Historical, Parody, Samurai, S…     TV       51
…                                                   …    …      …
12289                                             Hentai    OVA        1
12290                                             Hentai    OVA        1
12291                                             Hentai    OVA        4
12292                                             Hentai    OVA        1
12293                                             Hentai  Movie        1

       rating  members
0        9.37   200630
1        9.26   793665
2        9.25   114262
3        9.17   673572
4        9.16   151266
…       …      …
12289    4.15      211
12290    4.28      183
12291    4.88      219
12292    4.98      175
12293    5.46      142

[12294 rows x 7 columns]
```

[19]:

```
genres = set(genre for sublist in anime_data['genre'] for genre in sublist)
for genre in genres:
    anime_data[genre] = anime_data['genre'].apply(lambda x: 1 if genre in x␣
 ↪else 0)
 #Drop unnecessary columns
anime_data.drop(['anime_id', 'name', 'genre', 'type', 'episodes', 'rating',␣
 ↪'members'], axis=1, inplace=True)

# Calculate cosine similarity between items (anime)
item_similarity = cosine_similarity(anime_data)

# Convert the cosine similarity matrix into a DataFrame
item_similarity_df = pd.DataFrame(item_similarity, index=anime_data.index,␣
 ↪columns=anime_data.index)

def get_similar_anime(anime_id, top_n=5):
    # Get similarity scores for the given anime
    similar_anime = item_similarity_df.loc[anime_id].
 ↪sort_values(ascending=False)[1:top_n+1]
    return similar_anime

# Example usage:
similar_anime = get_similar_anime(anime_id=60, top_n=5)
print(similar_anime)
```

```
3089    0.973329
3544    0.973329
4418    0.973329
5805    0.971825
0       0.971825
Name: 60, dtype: float64
```

[ ]: