Mini-Project

July 14, 2023

Jupyter Notebook for the Mini-Project

Case Study: MovieLens Data Analysis

0.0.1 Proposed Question: Do certain genre and tag combinations tend to be more popular while their average rating is above the average of all other combinations?

Disclaimer: Throughout the process, I will leverage part of the codes Dr. Porter and Dr. Altintas built up in the lecture (only parts like the Data Ingestion, Data Cleaning that operates DataFrames). Data Engineering: Step 1, Acquire Data

0.1 Download the Dataset

This Notebook uses a dataset from the MovieLens website. Here are the links to the data source and location: * **Data Source:** MovieLens web site (filename: ml-25m.zip) * **Location:** https://grouplens.org/datasets/movielens/25m/ Once the download completes, please ensure the data files are in a directory called **movielens** in the same folder this **Notebook** lives.

Data Engineering: Step 2A, Exploring Data

0.2 Data Ingestion, Cleaning

In this notebook, we will use three CSV files: ratings.csv, tags.csv, and movies.csv

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

movies_df = pd.read_csv('./movielens/movies.csv', sep=',')
tags_df = pd.read_csv('./movielens/tags.csv', sep=',')
ratings_df = pd.read_csv('./movielens/ratings.csv', sep=',')

movies_df.head(), tags_df.head(), ratings_df.head()
```

[1]:
```
(   movieId                               title  \
 0        1                    Toy Story (1995)
```

```
1        2                      Jumanji (1995)
2        3             Grumpier Old Men (1995)
3        4            Waiting to Exhale (1995)
4        5  Father of the Bride Part II (1995)

                                       genres
0  Adventure|Animation|Children|Comedy|Fantasy
1                   Adventure|Children|Fantasy
2                              Comedy|Romance
3                        Comedy|Drama|Romance
4                                      Comedy  ,
   userId  movieId              tag   timestamp
0       3      260            classic  1439472355
1       3      260             sci-fi  1439472256
2       4     1732       dark comedy  1573943598
3       4     1732    great dialogue  1573943604
4       4     7569  so bad it's good  1573943455,
   userId  movieId  rating   timestamp
0       1      296     5.0  1147880044
1       1      306     3.5  1147868817
2       1      307     5.0  1147868828
3       1      665     5.0  1147878820
4       1      899     3.5  1147868510)
```

[2]:
```
# Drop unnecessary columns
movies_df.drop('title', axis=1, inplace=True)
tags_df.drop(['userId', 'timestamp'], axis=1, inplace=True)
ratings_df.drop(['userId', 'timestamp'], axis=1, inplace=True)

movies_df.head(), tags_df.head(), ratings_df.head()
```

[2]:
```
(   movieId                                       genres
 0        1  Adventure|Animation|Children|Comedy|Fantasy
 1        2                   Adventure|Children|Fantasy
 2        3                              Comedy|Romance
 3        4                        Comedy|Drama|Romance
 4        5                                      Comedy,
    movieId              tag
 0      260          classic
 1      260           sci-fi
 2     1732      dark comedy
 3     1732   great dialogue
 4     7569  so bad it's good,
    movieId  rating
 0      296     5.0
 1      306     3.5
 2      307     5.0
```

```
3      665     5.0
4      899     3.5)
```

Data Engineering: Step 2B, Pre-Processing Data

[3]:
```
# Check for missing values
movies_df.isnull().any(), tags_df.isnull().any(), ratings_df.isnull().any()
```

[3]:
```
(movieId    False
 genres     False
 dtype: bool,
 movieId    False
 tag         True
 dtype: bool,
 movieId    False
 rating     False
 dtype: bool)
```

[4]:
```
# Remove any missing values from the tags_df
tags_df.dropna(inplace=True)

tags_df.isnull().any()
```

[4]:
```
movieId    False
tag        False
dtype: bool
```

0.3 Data Merging

[5]:
```
# Calculate the average rating for each movie and overall
average_ratings_df = ratings_df.groupby('movieId')['rating'].mean().
 ↪reset_index()
average_rating = ratings_df['rating'].mean()

average_ratings_df, average_rating
```

[5]:
```
(       movieId    rating
 0            1  3.893708
 1            2  3.251527
 2            3  3.142028
 3            4  2.853547
 4            5  3.058434
 …          …        …
 59042   209157  1.500000
 59043   209159  3.000000
 59044   209163  4.500000
```

```
59045   209169  3.000000
59046   209171  3.000000

[59047 rows x 2 columns],
3.533854451353085)
```

[6]:
```
# Merge tags_df and movies_df
merged_df = pd.merge(tags_df, movies_df, on='movieId', how='inner')

# Combine the tags for each movie
merged_df = merged_df.groupby(['movieId', 'genres'])['tag'].agg('|'.join).
↪reset_index()

merged_df
```

[6]:
```
       movieId                                       genres  \
0            1  Adventure|Animation|Children|Comedy|Fantasy
1            2                   Adventure|Children|Fantasy
2            3                               Comedy|Romance
3            4                        Comedy|Drama|Romance
4            5                                      Comedy
…          …                                           …
45246   208813                                    Children
45247   208933                                      Horror
45248   209035                           Animation|Comedy
45249   209037                         (no genres listed)
45250   209063                         (no genres listed)

                                                     tag
0      Owned|imdb top 250|Pixar|Pixar|time travel|chi…
1      Robin Williams|time travel|fantasy|based on ch…
2      funny|best friend|duringcreditsstinger|fishing…
3      based on novel or book|chick flick|divorce|int…
4      aging|baby|confidence|contraception|daughter|g…
…                                                    …
45246                                       might like
45247               black and white|deal with the devil
45248  computer animation|Japan|mass behavior|mass sc…
45249  chameleon|computer animation|gluttony|humorous…
45250  black|education|friends schools|independent sc…

[45251 rows x 3 columns]
```

[7]:
```
# Merge with the average ratings dataframe and remove duplicates
merged_df = pd.merge(merged_df, average_ratings_df, on='movieId', how='inner')
merged_df.drop_duplicates(subset='movieId', inplace=True)
```

```
merged_df
```

[7]:
```
       movieId                                       genres  \
0            1  Adventure|Animation|Children|Comedy|Fantasy
1            2                   Adventure|Children|Fantasy
2            3                               Comedy|Romance
3            4                         Comedy|Drama|Romance
4            5                                       Comedy
…          …                                            …
41870   208813                                     Children
41871   208933                                       Horror
41872   209035                              Animation|Comedy
41873   209037                            (no genres listed)
41874   209063                            (no genres listed)

                                                      tag    rating
0      Owned|imdb top 250|Pixar|Pixar|time travel|chi…  3.893708
1      Robin Williams|time travel|fantasy|based on ch…  3.251527
2      funny|best friend|duringcreditsstinger|fishing…  3.142028
3      based on novel or book|chick flick|divorce|int…  2.853547
4      aging|baby|confidence|contraception|daughter|g…  3.058434
…                                                    …       …
41870                                        might like  3.000000
41871                black and white|deal with the devil  2.500000
41872  computer animation|Japan|mass behavior|mass sc…  3.500000
41873  chameleon|computer animation|gluttony|humorous…  4.000000
41874  black|education|friends schools|independent sc…  4.000000

[41875 rows x 4 columns]
```

0.4 Data Filtering

[8]:
```
# Filter out movies with no listed genres and movies with a rating less than␣
 ↪the average rating
merged_df = merged_df[(merged_df['genres'] != '(no genres listed)') &␣
 ↪(merged_df['rating'] > average_rating)]

# Reset index
merged_df.reset_index(drop=True, inplace=True)

merged_df
```

[8]:
```
      movieId                                       genres  \
0           1  Adventure|Animation|Children|Comedy|Fantasy
1           6                         Action|Crime|Thriller
```

```
2          11                          Comedy|Drama|Romance
3          16                                  Crime|Drama
4          17                               Drama|Romance
…         …                                           …
9788   208465                 Action|Crime|Drama|Thriller
9789   208591                                 Documentary
9790   208735                           Documentary|War
9791   208747                             Drama|Mystery
9792   208800                            Comedy|Romance

                                                    tag    rating
0     Owned|imdb top 250|Pixar|Pixar|time travel|chi…  3.893708
1     imdb top 250|great acting|realistic action|sus…  3.854909
2     Romance|white house|new love|usa president|whi…  3.657171
3     Mafia|Mafia|Martin Scorsese|organized crime|ro…  3.823707
4     chick flick|British|Jane Austen|19th century|a…  3.948806
…                                                  …       …
9788  chaos|Direction|screenplay|sound effects|story…  4.500000
9789                                       might like  4.000000
9790              documentary|interviews|memory|Panama  4.000000
9791                                       might like  3.750000
9792                                       might like  3.625000

[9793 rows x 4 columns]
```

0.5 Vectorized String Operations

[9]:
```
# Split 'genres' and 'tag' into lists
merged_df = merged_df.copy()
merged_df['genres'] = merged_df['genres'].str.split('|')
merged_df['tag'] = merged_df['tag'].str.split('|')

merged_df
```

[9]:
```
      movieId                                             genres  \
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
1           6                          [Action, Crime, Thriller]
2          11                           [Comedy, Drama, Romance]
3          16                                    [Crime, Drama]
4          17                                 [Drama, Romance]
…         …                                                 …
9788   208465                  [Action, Crime, Drama, Thriller]
9789   208591                                     [Documentary]
9790   208735                                [Documentary, War]
9791   208747                                  [Drama, Mystery]
9792   208800                                 [Comedy, Romance]
```

```
                                                    tag    rating
0     [Owned, imdb top 250, Pixar, Pixar, time trave…  3.893708
1     [imdb top 250, great acting, realistic action,…  3.854909
2     [Romance, white house, new love, usa president…  3.657171
3     [Mafia, Mafia, Martin Scorsese, organized crim…  3.823707
4     [chick flick, British, Jane Austen, 19th centu…  3.948806
…                                                   …       …
9788  [chaos, Direction, screenplay, sound effects, …  4.500000
9789                                     [might like]  4.000000
9790          [documentary, interviews, memory, Panama]  4.000000
9791                                     [might like]  3.750000
9792                                     [might like]  3.625000

[9793 rows x 4 columns]
```

Data Analysis: Step 3, Analyze Data

0.6 Data Transformation and Feature Engineering

[10]:
```
from itertools import product

# Generate all possible combinations of genres and tags for each movie
merged_df['genre_tag'] = merged_df.apply(lambda row:␣
 ↪list(product(row['genres'], row['tag'])), axis=1)

merged_df
```

[10]:
```
      movieId                                              genres  \
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
1           6                           [Action, Crime, Thriller]
2          11                            [Comedy, Drama, Romance]
3          16                                      [Crime, Drama]
4          17                                   [Drama, Romance]
…         …                                                   …
9788   208465                  [Action, Crime, Drama, Thriller]
9789   208591                                       [Documentary]
9790   208735                                  [Documentary, War]
9791   208747                                    [Drama, Mystery]
9792   208800                                   [Comedy, Romance]

                                                    tag    rating  \
0     [Owned, imdb top 250, Pixar, Pixar, time trave…  3.893708
1     [imdb top 250, great acting, realistic action,…  3.854909
2     [Romance, white house, new love, usa president…  3.657171
3     [Mafia, Mafia, Martin Scorsese, organized crim…  3.823707
```

```
4     [chick flick, British, Jane Austen, 19th centu…  3.948806
…                                                   …       …
9788  [chaos, Direction, screenplay, sound effects, …  4.500000
9789                                     [might like]  4.000000
9790          [documentary, interviews, memory, Panama]  4.000000
9791                                     [might like]  3.750000
9792                                     [might like]  3.625000

                                                genre_tag
0     [(Adventure, Owned), (Adventure, imdb top 250)…
1     [(Action, imdb top 250), (Action, great acting…
2     [(Comedy, Romance), (Comedy, white house), (Co…
3     [(Crime, Mafia), (Crime, Mafia), (Crime, Marti…
4     [(Drama, chick flick), (Drama, British), (Dram…
…                                                   …
9788  [(Action, chaos), (Action, Direction), (Action…
9789                       [(Documentary, might like)]
9790  [(Documentary, documentary), (Documentary, int…
9791       [(Drama, might like), (Mystery, might like)]
9792      [(Comedy, might like), (Romance, might like)]

[9793 rows x 5 columns]
```

[11]:
```
# Explode the dataframe on the 'genre_tag' column
exploded_df = merged_df.explode('genre_tag')

exploded_df
```

[11]:
```
      movieId                                          genres  \
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
…         …                                                  …
9790   208735                                [Documentary, War]
9791   208747                                  [Drama, Mystery]
9791   208747                                  [Drama, Mystery]
9792   208800                                 [Comedy, Romance]
9792   208800                                 [Comedy, Romance]

                                                    tag    rating  \
0     [Owned, imdb top 250, Pixar, Pixar, time trave…  3.893708
0     [Owned, imdb top 250, Pixar, Pixar, time trave…  3.893708
0     [Owned, imdb top 250, Pixar, Pixar, time trave…  3.893708
0     [Owned, imdb top 250, Pixar, Pixar, time trave…  3.893708
0     [Owned, imdb top 250, Pixar, Pixar, time trave…  3.893708
```

```
…                                                     …       …
9790          [documentary, interviews, memory, Panama]  4.000000
9791                                          [might like]  3.750000
9791                                          [might like]  3.750000
9792                                          [might like]  3.625000
9792                                          [might like]  3.625000

                     genre_tag
0            (Adventure, Owned)
0     (Adventure, imdb top 250)
0            (Adventure, Pixar)
0            (Adventure, Pixar)
0      (Adventure, time travel)
…                          …
9790             (War, Panama)
9791       (Drama, might like)
9791     (Mystery, might like)
9792      (Comedy, might like)
9792     (Romance, might like)

[1674005 rows x 5 columns]
```

[12]:
```
# Split the 'genre_tag' column into two separate columns
exploded_df[['genre', 'tag']] = pd.DataFrame(exploded_df['genre_tag'].tolist(),␣
 ↪index=exploded_df.index)

# Drop the 'genre_tag' column
exploded_df = exploded_df.drop('genre_tag', axis=1)

exploded_df
```

[12]:
```
      movieId                                              genres  \
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
0           1  [Adventure, Animation, Children, Comedy, Fantasy]
…       …                                                    …
9790   208735                                 [Documentary, War]
9791   208747                                   [Drama, Mystery]
9791   208747                                   [Drama, Mystery]
9792   208800                                  [Comedy, Romance]
9792   208800                                  [Comedy, Romance]

               tag    rating      genre
0            Owned  3.893708  Adventure
0     imdb top 250  3.893708  Adventure
```

```
0            Pixar  3.893708  Adventure
0            Pixar  3.893708  Adventure
0      time travel  3.893708  Adventure
…              …       …         …
9790        Panama  4.000000        War
9791    might like  3.750000      Drama
9791    might like  3.750000    Mystery
9792    might like  3.625000     Comedy
9792    might like  3.625000    Romance

[1674005 rows x 5 columns]
```

[13]:
```
# Calculate the total number of ratings and the average rating for each␣
↪genre-tag combination
genre_tag_df = exploded_df.groupby(['genre', 'tag']).agg({'rating': ['count',␣
↪'mean']}).reset_index()

# Flatten the column names
genre_tag_df.columns = ['genre', 'tag', 'rating_count', 'rating_mean']

genre_tag_df
```

[13]:
```
          genre                              tag  rating_count  rating_mean
0        Action            "Nut up or shut up"             2     3.762951
1        Action                  "The Hunter"              2     3.690157
2        Action  "We're on a mission from god"             1     3.800171
3        Action              "damn dirty apes"             2     3.625190
4        Action          "retrofitted" future              2     4.120189
…         …                                …           …          …
163082  Western                    wyatt earp             2     3.785510
163083  Western                       wyoming             5     3.921146
163084  Western                younger brother             1     3.600000
163085  Western                       youtube             1     3.819149
163086  Western                zooey deschanel             1     3.661401

[163087 rows x 4 columns]
```

[14]:
```
# Calculate the overall average rating count and mean rating
average_rating_count = genre_tag_df['rating_count'].mean()
average_rating_mean = genre_tag_df['rating_mean'].mean()

average_rating_mean
```

[14]:
```
3.815040997340062
```

[15]:
```
# Identify genre-tag combinations with a higher total number of ratings and a␣
↪higher average rating than the overall average
```

```
popular_high_rated_combinations = genre_tag_df[(genre_tag_df['rating_count'] >␣
↪average_rating_count) & (genre_tag_df['rating_mean'] > average_rating_mean)]

popular_high_rated_combinations
```

[15]:

	genre	tag	rating_count	rating_mean
79	Action	1930s	13	3.872918
80	Action	1940s	14	3.887185
85	Action	1970s	40	3.821734
94	Action	19th century	34	3.925425
131	Action	4th wall	95	3.821347
…	…	…	…	…
163007	Western	violence	102	3.932458
163009	Western	violent	48	3.926776
163015	Western	visually appealing	178	3.962695
163024	Western	war	16	3.830560
163045	Western	western	368	3.911015

[10608 rows x 4 columns]

[16]:
```
# Sort the dataframe by rating count in descending order
popular_high_rated_combinations = popular_high_rated_combinations.copy()
popular_high_rated_combinations.sort_values('rating_count', ascending=False,␣
↪inplace=True)

# Reset the index of the dataframe
popular_high_rated_combinations.reset_index(drop=True, inplace=True)

popular_high_rated_combinations
```

[16]:

	genre	tag	rating_count	rating_mean
0	Sci-Fi	sci-fi	5655	3.894882
1	Drama	atmospheric	3686	3.925056
2	Action	sci-fi	3567	3.902858
3	Drama	surreal	3107	3.896263
4	Thriller	twist ending	3075	3.985348
…	…	…	…	…
10603	Thriller	use of music	11	3.984191
10604	Drama	weightlessness	11	4.155508
10605	Drama	wildly overrated	11	4.084434
10606	Comedy	United Kingdom	11	3.996194
10607	Sci-Fi	excessive violence	11	3.938141

[10608 rows x 4 columns]

[17]:
```
# Display the top 10 genre-tag combinations for visualization
top_10_combinations = popular_high_rated_combinations.head(10)
```

```
top_10_combinations
```

[17]:

	genre	tag	rating_count	rating_mean
0	Sci-Fi	sci-fi	5655	3.894882
1	Drama	atmospheric	3686	3.925056
2	Action	sci-fi	3567	3.902858
3	Drama	surreal	3107	3.896263
4	Thriller	twist ending	3075	3.985348
5	Action	action	3019	3.868889
6	Comedy	comedy	2710	3.825872
7	Drama	twist ending	2696	3.961640
8	Adventure	sci-fi	2590	3.893345
9	Mystery	twist ending	2577	3.973027

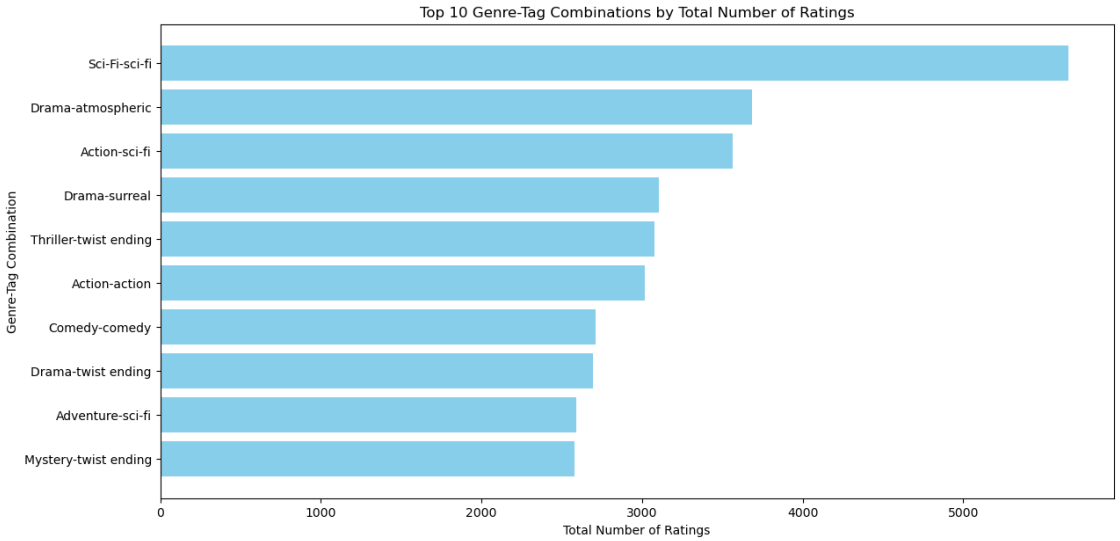
[18]:
```
# Sort the top 10 combinations by average rating in descending order
top_10_combinations_sorted = top_10_combinations.sort_values('rating_mean',␣
↪ascending=False)

top_10_combinations_sorted
```

[18]:

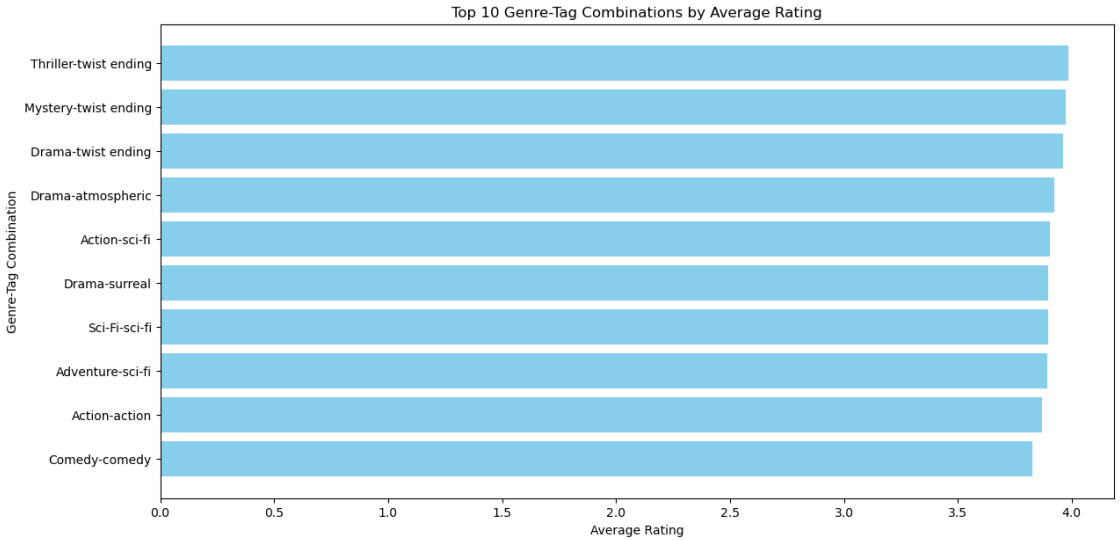
	genre	tag	rating_count	rating_mean
4	Thriller	twist ending	3075	3.985348
9	Mystery	twist ending	2577	3.973027
7	Drama	twist ending	2696	3.961640
1	Drama	atmospheric	3686	3.925056
2	Action	sci-fi	3567	3.902858
3	Drama	surreal	3107	3.896263
0	Sci-Fi	sci-fi	5655	3.894882
8	Adventure	sci-fi	2590	3.893345
5	Action	action	3019	3.868889
6	Comedy	comedy	2710	3.825872

Data Analysis: Step 4, Reporting Insights

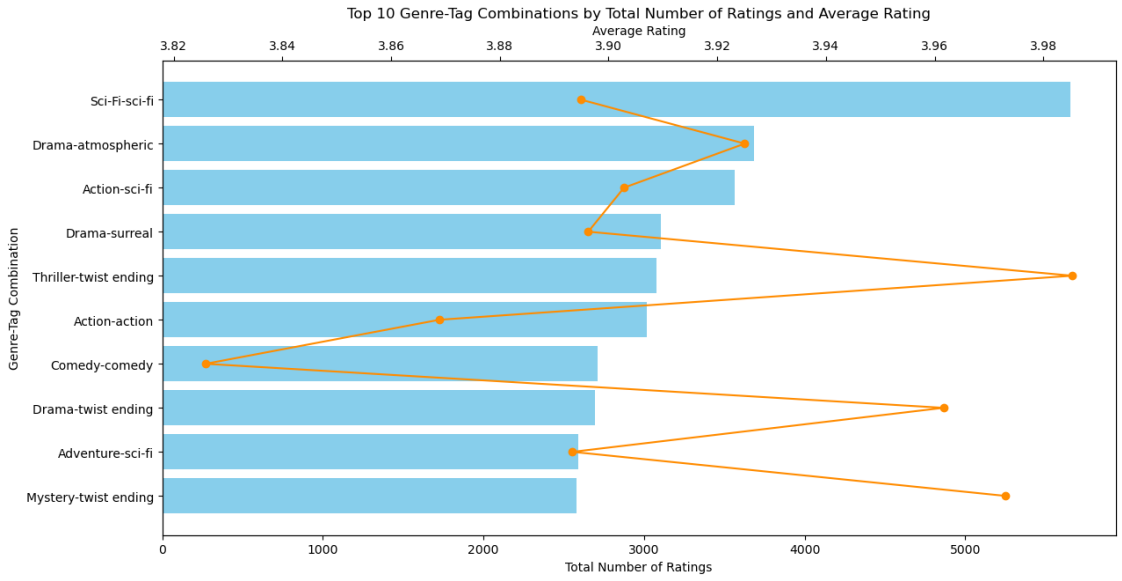
[19]:
```
# Create the bar plot for rating counts
plt.figure(figsize=(14, 7))
plt.barh(top_10_combinations['genre'] + '-' + top_10_combinations['tag'],␣
↪top_10_combinations['rating_count'], color='skyblue')
plt.xlabel('Total Number of Ratings')
plt.ylabel('Genre-Tag Combination')
plt.title('Top 10 Genre-Tag Combinations by Total Number of Ratings')
plt.gca().invert_yaxis()
plt.show()
```



[20]:
```
# Create the bar plot for average ratings
plt.figure(figsize=(14, 7))
plt.barh(top_10_combinations_sorted['genre'] + '-' +␣
 ↪top_10_combinations_sorted['tag'],␣
 ↪top_10_combinations_sorted['rating_mean'], color='skyblue')
plt.xlabel('Average Rating')
plt.ylabel('Genre-Tag Combination')
plt.title('Top 10 Genre-Tag Combinations by Average Rating')
plt.gca().invert_yaxis()
plt.show()
```



[21]:
```
# Merge top_10_combinations and top_10_combinations_sorted dataframes on genre␣
↪and tag
combined_df = pd.merge(top_10_combinations, top_10_combinations_sorted,␣
↪on=['genre', 'tag'], suffixes=('_count', '_mean'))
# Flip the dataframe
combined_df_flipped = combined_df[::-1]

fig, ax1 = plt.subplots(figsize=(14, 7))
ax1.barh(combined_df_flipped['genre'] + '-' + combined_df_flipped['tag'],␣
↪combined_df_flipped['rating_count_count'], color='skyblue')
ax1.set_xlabel('Total Number of Ratings')
ax1.set_ylabel('Genre-Tag Combination')
# Create a second y-axis to plot average ratings
ax2 = ax1.twiny()
ax2.plot(combined_df_flipped['rating_mean_mean'], combined_df_flipped['genre']␣
↪+ '-' + combined_df_flipped['tag'], color='darkorange', marker='o')
ax2.set_xlabel('Average Rating')
plt.title('Top 10 Genre-Tag Combinations by Total Number of Ratings and Average␣
↪Rating')
plt.show()
```



[22]:
```
# # Calculate average rating and number of ratings for each movie
# average_ratings = ratings_df.groupby('movieId')['rating'].mean()
# num_ratings = ratings_df.groupby('movieId').size()

# # Create a new dataframe with the calculated values
```

```
# ratings_summary = pd.DataFrame({'average_rating': average_ratings,␣
 ↪'num_ratings': num_ratings})

# # Reset index to make 'movieId' a column for merging
# ratings_summary.reset_index(inplace=True)

# # Merge the ratings_summary dataframe with the movies dataframe
# movies_ratings = pd.merge(movies_df, ratings_summary, on='movieId')

# # Create a scatter plot of 'num_ratings' vs 'average_rating'
# plt.figure(figsize=(10, 6))
# plt.scatter(movies_ratings['num_ratings'], movies_ratings['average_rating'],␣
 ↪alpha=0.5)
# plt.title('Number of Ratings vs Average Rating for Movies')
# plt.xlabel('Number of Ratings')
# plt.ylabel('Average Rating')
# plt.show()
```