Assignment 8

August 24, 2023

1 Assignment 8

1.0.1 Joseph Annand

1.0.2 8/25/2023

1.0.3 The libraries you will use are already loaded for you below

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

1.1 Question 1

Read in the two Netflix CSV files from /Data/Netflix as pandas dataframes. Print the number of unique genres. This is not as simple as it sounds. You cannot simply find the length of `titles['genres'].unique()`. You must convert the output of that code to a list, iterate over that list and replace the following characters: `[]',`. Once you have them replace you can split the individual strings to list items and flatten the list. I have already imported the `chain()` function for you to flatten the list. Look up the documentation to see its usage. There are 19 unique genres, but I want you to write the code to find them.

[2]:
```
# your code here
titles_data = pd.read_csv("Data/Netflix/titles.csv")
credits_data = pd.read_csv("Data/Netflix/credits.csv")

# Create list of unqiue values from genre column
genres = list(titles_data['genres'].unique())

# Replace [],' characters and split values with multiple genres into lists
for i in range(len(genres)):
    genres[i] = genres[i].replace("[", "").replace("]", "").replace(",", "").
 ↪replace("'", "")
    genres[i] = genres[i].split()

# Flatten the list of genres
genres = list(chain.from_iterable(genres))
```

```
# Get unique values for genres
genres = pd.Series(genres).unique()
# Print size of genres series
print(genres.size)
```

```
19
```

1.2 Question 2

Print the release year and the imdb score of the highest average score of all movies by year. This is trickier than it sounds. To do this you will need to aggregate the means by year. If you use the simple method you will get a pandas series. The series will need to be converted to a dataframe and the index will need to be set as a column (release year). Once you have done that you can find the numerical index with the highest average imdb score.

[3]:
```
# your code here
# Filter out the TV shows
movies = titles_data[titles_data['type'] == "MOVIE"]

# Group movies by release year and get maximum imdb score for each
movies_year = movies.groupby(by="release_year")['imdb_score'].max()

# Convert movie_year Series to data frame
highest_rated = pd.DataFrame(movies_year)

# Print the index of the highest rated movie
print(highest_rated.idxmax())
```

```
imdb_score    1979
dtype: int64
```

1.3 Question 3

There were 208 actors in the movie with the most credited actors. What is the title of that movie? Nulls and NaN values do not count.

[4]:
```
# your code here
# Group credits_data by movie id and get count of all values with that id
acting_credits = pd.DataFrame(credits_data.groupby(by='id')['name'].count())
# Get value of movie id with the most credits
movie_id = acting_credits['name'].idxmax()
# Get title of movie that matches the id with the most credits
print(pd.DataFrame(titles_data.loc[titles_data['id'] == movie_id, 'title']))
```

```
              title
718  Les Misérables
```

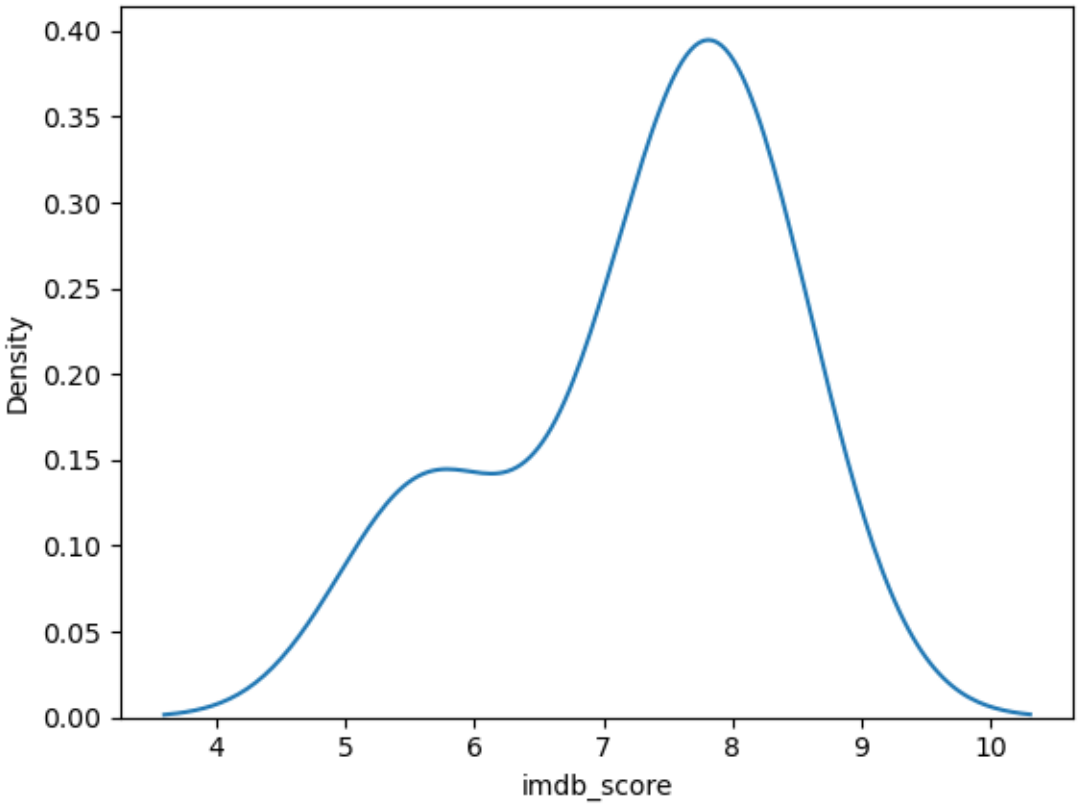
1.4 Question 4

Which movie has the highest IMDB score for the actor Robert De Niro? What year was it made? Create a kdeplot (kernel density estimation to show the distribution of his IMDB movie scores.

[5]:
```
# your code here
# Get all credits for Robert De Niro
de_niro_credits = pd.DataFrame(credits_data.loc[credits_data['name'] == "Robert␣
↪De Niro", 'id'])

# Create list of each id for movies featuring Robert De Niro
de_niro_id = list(de_niro_credits['id'])

# Find all entires that match the De Niro movie ids
de_niro_movies = pd.DataFrame(titles_data.loc[titles_data['id'].
↪isin(de_niro_id), :])

# Print the title and release year of the movie that has the highest imdb score
print(de_niro_movies.at[de_niro_movies['imdb_score'].idxmax(), 'title'])
print(de_niro_movies.at[de_niro_movies['imdb_score'].idxmax(), 'release_year'])

# Create kernel density plot of imdb scores for De Niro movies
sns.kdeplot(data=de_niro_movies, x="imdb_score")
```

```
Taxi Driver
1976
```

[5]:
```
<Axes: xlabel='imdb_score', ylabel='Density'>
```



1.5 Question 5

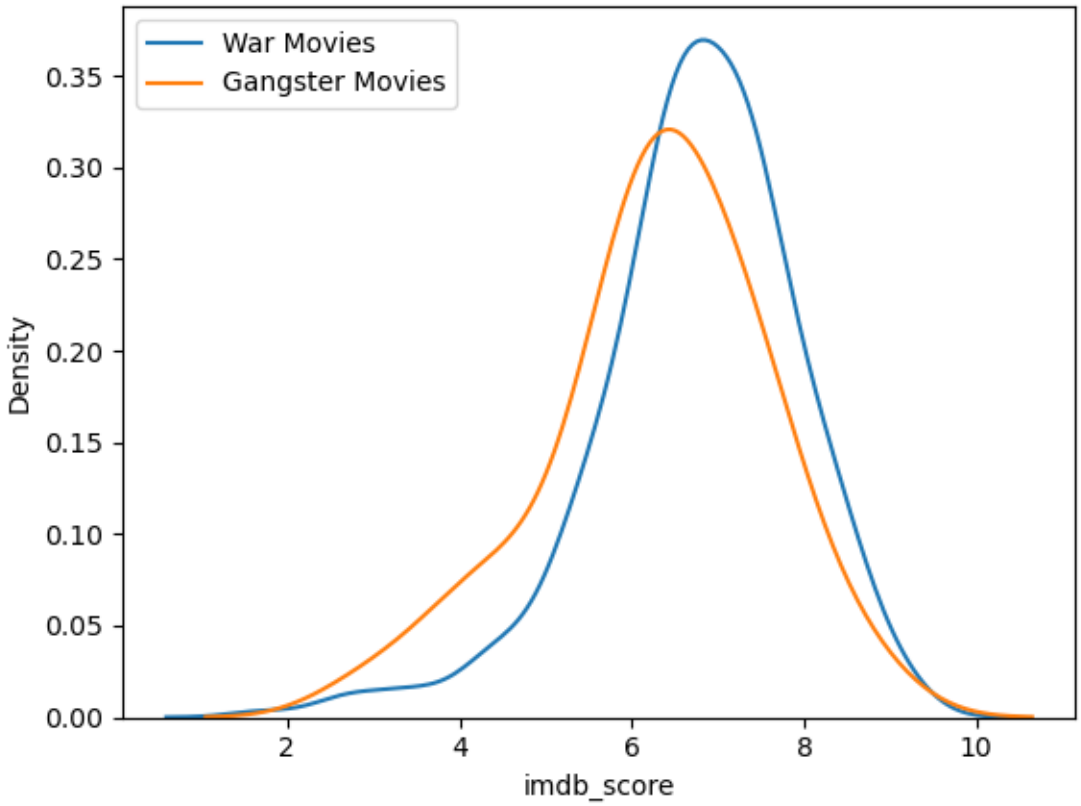
Create two new boolean columns in the titles dataframe that are true when the description contains war or gangster. Call these columns `war_movies` and `gangster_movies`. How many movies are there in both categories? Which category has a higher average IMDB score? Show the IMDB score kernel density estimations of both categories.

[14]:
```
# your code here
# Create bool column that shows if movie description mentions "war"
titles_data['war_movies'] = np.where(titles_data['description'].str.
 ↪contains("war"), True, False)
# Create bool column that shows if movie description mentions "gangster"
titles_data['gangster_movies'] = np.where(titles_data['description'].str.
 ↪contains("gangster"), True, False)

# Find how many movies mention "war"
print(len(titles_data[titles_data['war_movies'] == True]))
# Find how many movies mention "gangster"
print(len(titles_data[titles_data['gangster_movies'] == True]))

# Get data for war movies
```

```
war_movie_data = titles_data.loc[titles_data['war_movies'] == True, :]
# Get data for gangster movies
gang_movie_data = titles_data.loc[titles_data['gangster_movies'] == True, :]

# Calculate mean imdb score for war movies
war_score = np.mean(war_movie_data['imdb_score'])
# Calculate mean imdb score for gangster movies
gangster_score = np.mean(gang_movie_data['imdb_score'])

# Determine which average score is higher
if  (war_score > gangster_score):
    print(f"Average war movie IMDb score of {war_score} is higher than average␣
 ↪gangster movie score of {gangster_score}")
else:
    print(f"Average ganster movie IMDb score of {gangster_score} is higher than␣
 ↪average war movie IMDb score of {war_score}")

# Create kernel density plot for imdb scores of war movies
sns.kdeplot(data=war_movie_data, x='imdb_score', label='War Movies')
# Create kernel density plot for imdb scores of gangster movies
sns.kdeplot(data=gang_movie_data, x='imdb_score', label='Gangster Movies')
plt.legend()
```

```
384
51
Average war movie IMDb score of 6.738728323699423 is higher than average
gangster movie score of 6.276315789473683
```

[14]: <matplotlib.legend.Legend at 0x1d86c583d30>


War Movies
Gangster Movies
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Density
2
4
6
8
10
imdb_score

[ ]: