Assignment 8

June 11, 2023

1 Assignment 8

1.0.1 Sue Susman MEd, BSN, RN

1.0.2 Date: 06-11-2023

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

[4]:
```
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.

[23]:
```
# your code here
df1 = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class Files/DSE␣
↪5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/titles.csv')
df2 = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class Files/DSE␣
↪5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/credits.csv')

combined_df = pd.concat([df1, df2], ignore_index=True)

genres = combined_df['genres'].unique().tolist()

# Remove NaN values from the list
genres = [genre for genre in genres if isinstance(genre, str)]

genres = [genre.replace('[', '').replace(']', '').replace("'", '').replace(',',␣
↪'') for genre in genres]
```

```
flattened_genres = list(chain(*[genre.split() for genre in genres]))

unique_genres = len(set(flattened_genres))
print("Number of unique genres:", unique_genres)
```

```
Number of unique genres: 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.

[41]:
```
# your code here
import pandas as pd

# Read the Netflix CSV file into a pandas DataFrame
df1 = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class Files/DSE␣
 ↪5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/titles.csv')
df2 = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class Files/DSE␣
 ↪5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/credits.csv')

# Convert the 'release_year' column to numeric
df1['release_year'] = pd.to_numeric(df1['release_year'], errors='coerce')

# Group the DataFrame by 'release_year' and calculate the mean IMDb score for␣
 ↪each year
grouped_df1 = df1.groupby('release_year')['imdb_score'].mean()

# Convert the resulting Series to a DataFrame and reset the index
grouped_df1 = grouped_df1.reset_index()

# Find the row with the highest average IMDb score
highest_average = grouped_df1.loc[grouped_df1['imdb_score'].idxmax()]

# Extract the release year and IMDb score from the row
release_year = int(highest_average['release_year'])
average_imdb_score = highest_average['imdb_score']

# Print the release year and IMDb score of the highest average score
print("Release Year:", release_year)
print("Highest Average IMDb Score:", average_imdb_score)
```

```
Release Year: 1985
Highest Average IMDb Score: 8.0
```

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.

[15]:
```
import pandas as pd

# Read the Netflix CSV files containing credits and titles into pandas DataFrames
credits_df = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class␣
 ↪Files/DSE 5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/
 ↪credits.csv')
titles_df = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class Files/
 ↪DSE 5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/titles.csv')

# Merge the credits and titles DataFrames based on the 'show_id' column
merged_df = pd.merge(credits_df, titles_df, on ='id')

# Drop rows with null or NaN values in the 'cast' column
merged_df = merged_df.dropna(subset=['cast'])

# Count the number of credited actors for each movie
merged_df['num_actors'] = merged_df['cast'].apply(lambda x: len(x.split(',')))

# Find the movie with the highest number of credited actors
max_actors_movie = merged_df.loc[merged_df['num_actors'].idxmax()]

# Extract the title of the movie with the most credited actors
movie_title = max_actors_movie['title']

# Print the title of the movie
print("Title of the movie with the most credited actors:", movie_title)
```

```
Title of the movie with the most credited actors: Dance & Sing With True
```

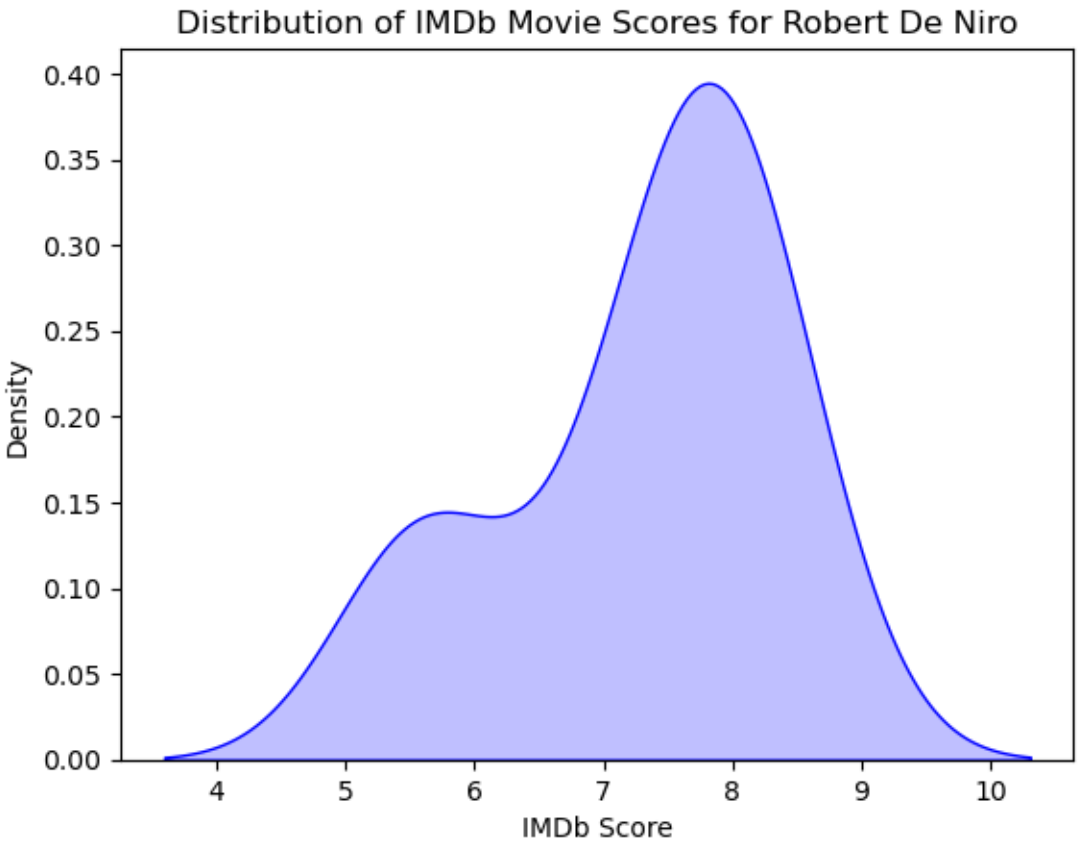
1.4 Question 4

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.

[16]:
```
# your code here
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Read the Netflix CSV files containing credits and titles into pandas DataFrames
credits_df = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class␣
 ↪Files/DSE 5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/
 ↪credits.csv')
```

```
titles_df = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class Files/
 ↪DSE 5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/titles.csv')

# Merge the credits and titles DataFrames based on the 'id' column
merged_df = pd.merge(credits_df, titles_df, on='id')

# Filter the DataFrame to include only rows where Robert De Niro appears in the␣
 ↪'cast' column
robert_de_niro_df = merged_df[merged_df['cast'].str.contains('Robert De Niro')]

# Find the movie with the highest IMDb score for Robert De Niro
highest_score_movie = robert_de_niro_df.loc[robert_de_niro_df['imdb_score'].
 ↪idxmax()]

# Extract the title and year of the movie
movie_title = highest_score_movie['title']
release_year = highest_score_movie['release_year']

# Print the title and year of the movie
print("Movie with the highest IMDb score for Robert De Niro:")
print("Title:", movie_title)
print("Year of Release:", release_year)

# Create a KDE plot to show the distribution of Robert De Niro's IMDb movie␣
 ↪scores
sns.kdeplot(data=robert_de_niro_df, x='imdb_score', fill=True, color='blue')
plt.title("Distribution of IMDb Movie Scores for Robert De Niro")
plt.xlabel("IMDb Score")
plt.ylabel("Density")
plt.show()
```

```
Movie with the highest IMDb score for Robert De Niro:
Title: Taxi Driver
Year of Release: 1976
```



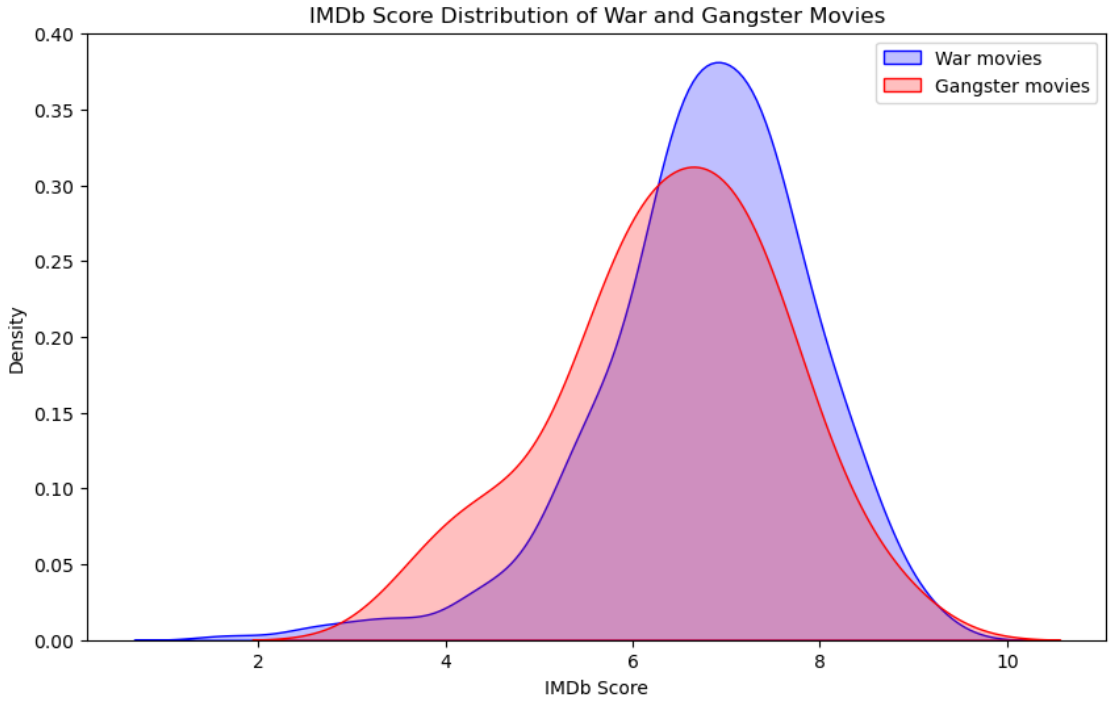
1.5 Question 5

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.

[19]:
```
# your code here
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Read the Netflix titles CSV file into a pandas DataFrame
titles_df = pd.read_csv('/Users/sue_susman/Desktop/2023 Data Science Class Files/
 ↪DSE 5002 R & Python Programming/GitHub/DSE5002/Week_8/Data/Netflix/titles.csv')

# Create the 'war_movies' column with boolean values indicating if the␣
 ↪description contains 'war'
titles_df['war_movies'] = titles_df['description'].str.contains('war',␣
 ↪case=False).fillna(False)
```

```
# Create the 'gangster_movies' column with boolean values indicating if the␣
 ↪description contains 'gangster'
titles_df['gangster_movies'] = titles_df['description'].str.contains('gangster',␣
 ↪case=False).fillna(False)

# Count the number of movies in both categories
num_war_movies = titles_df['war_movies'].sum()
num_gangster_movies = titles_df['gangster_movies'].sum()

# Print the number of movies in both categories
print("Number of movies in the 'war' category:", num_war_movies)
print("Number of movies in the 'gangster' category:", num_gangster_movies)

# Calculate the average IMDb score for both categories
avg_imdb_war = titles_df[titles_df['war_movies']]['imdb_score'].mean()
avg_imdb_gangster = titles_df[titles_df['gangster_movies']]['imdb_score'].mean()

# Print the average IMDb scores for both categories
print("Average IMDb score for 'war' movies:", avg_imdb_war)
print("Average IMDb score for 'gangster' movies:", avg_imdb_gangster)

# Create KDE plots to show the distribution of IMDb scores for 'war' and␣
 ↪'gangster' movies
plt.figure(figsize=(10, 6))
sns.kdeplot(data=titles_df[titles_df['war_movies']], x='imdb_score', fill=True,␣
 ↪color='blue', label='War movies')
sns.kdeplot(data=titles_df[titles_df['gangster_movies']], x='imdb_score',␣
 ↪fill=True, color='red', label='Gangster movies')
plt.title("IMDb Score Distribution of War and Gangster Movies")
plt.xlabel("IMDb Score")
plt.ylabel("Density")
plt.legend()
plt.show()
```

```
Number of movies in the 'war' category: 437
Number of movies in the 'gangster' category: 35
Average IMDb score for 'war' movies: 6.772439024390243
Average IMDb score for 'gangster' movies: 6.3914285714285715
```


IMDb Score Distribution of War and Gangster Movies
War movies
Gangster movies
Density
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
2
4
6
8
10
IMDb Score

[ ]: