Mia Friedman
Sarah Wojtowicz
Group Name: Ratings v.s. Price
December 12, 2022

Final Project Report

Link to github repository: https://github.com/sarahwoj/SI206_finalproject.git

1.The goals for your project (10 points)

The goal of our project was to successfully use and gather information from APIs and websites using BeautifulSoup to answer the question of whether movie budget has a correlation with its rating on either IMDB or Rotten Tomatoes and if so, what is the effect.

2.The goals that were achieved (10 points)

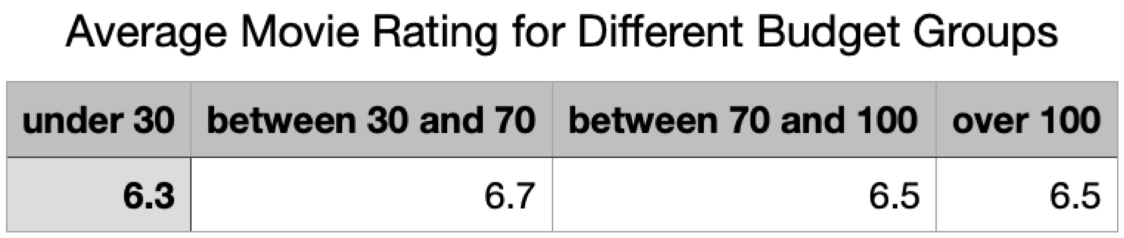
Overall, we were able to use two APIs to successfully achieve our goal of finding that there is little to no correlation between movie budget and movie rating. We successfully used BeautifulSoup to get a viable list of movies, and used that list with both APIs to get the respective IMDB movie ratings and budgets.

3.The problems that you faced (10 points)

We originally wanted to use BeautifulSoup on the Wikipedia pages of the movies in our movie list to get the budget. However, we ran into problems iterating through the list of movies to get the budget from each Wikipedia page since some of the Wikipedia page titles weren't the same as the title of the movie in our movie list. For example, the title of one of the movies in our list was "Scream" but its Wikipedia page name was "Scream (2022 Film)". For this reason, we searched for other options to get the movie budgets. We were able to find and use an API to get the movie budget data. However, the API didn't have the data for every movie in our list which is why some of the budget data points have a 0 rather than a viable budget. We had to take this into account when doing our calculations and visualizations to make sure to include code to factor out these data points from our calculations so they wouldn't skew our results.

4.Your file that contains the calculations from the data in the database (10 points)

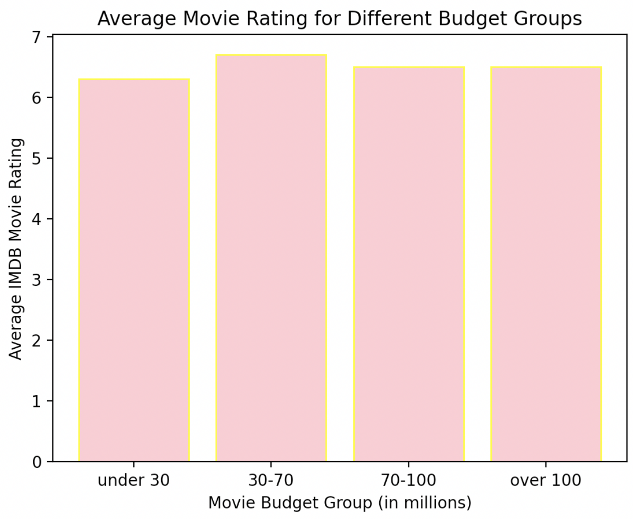
(csv file in project folder)

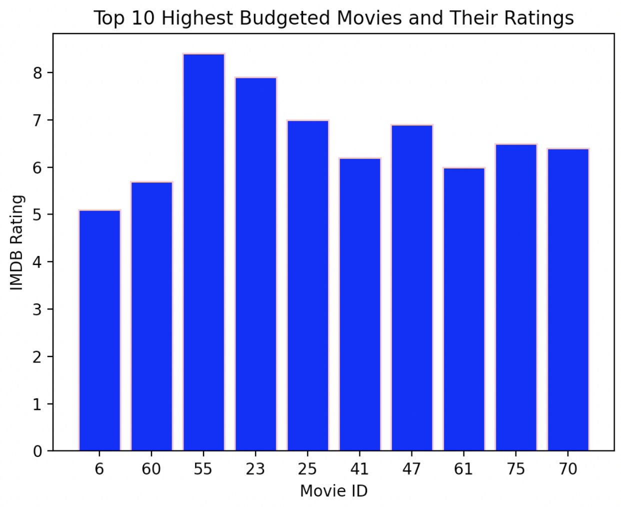


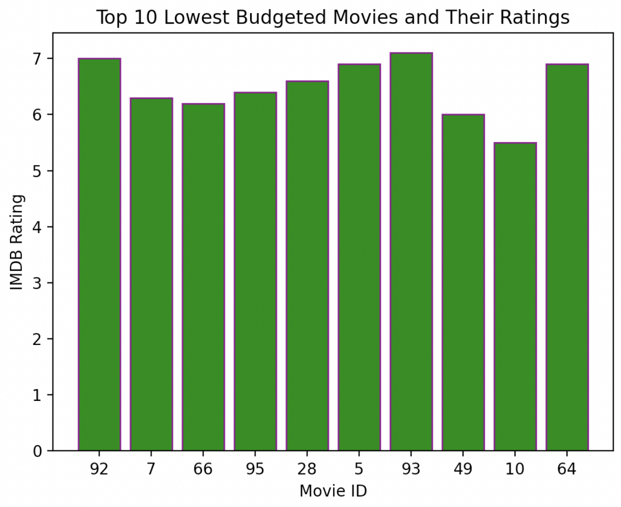
Average Movie Rating for Different Budget Groups

under 30	between 30 and 70	between 70 and 100	over 100
6.3	6.7	6.5	6.5

5.The visualization that you created (i.e. screen shot or image file) (10 points)


Average Movie Rating for Different Budget Groups
Average IMDB Movie Rating
Movie Budget Group (in millions)
under 30
30-70
70-100
over 100
0
1
2
3
4
5
6
7


Top 10 Highest Budgeted Movies and Their Ratings
IMDB Rating
Movie ID
6
60
55
23
25
41
47
61
75
70
0
1
2
3
4
5
6
7
8


Top 10 Lowest Budgeted Movies and Their Ratings
IMDB Rating
Movie ID
92
7
66
95
28
5
93
49
10
64
0
1
2
3
4
5
6
7

6.Instructions for running your code (10 points)

How to run our project code:

1. First, run the api python file to get the list of movies with beautiful soup and their respective IMDB ratings from the Open Movie Database API. The api file will then open the final_db database, create the Movie_ids and Ratings table and add the movie name, movie id and ratings information to the tables. Run this api file 4 times to add all 100 values to the tables, since the code will only add 25 items to the tables each time.
2. Then, run the bs python file to get the list of movies again with beautiful soup and their respective budgets from the Movie Database API. The bs file will then open the final_db database, create the Budgets table and add the Movie id and Budget information to the table. Run this api file 4 times to add all 100 values to the tables, since the code will only add 25 items to the tables each time.
3. After running both the api and bs python files 4 times each, run the calc python file to join the Ratings and Budget tables, sort the movies in different budget groups into lists, and calculate the average IMDB rating for all the movies in the budget group. It will then write a csv file that will show up in the project folder providing the average rating for the different budget groups.
4. Then you will run the visualizations python file to create 3 graphs visualizing the data. The first visualization is a bar graph showing the average rating for the different budget groups. The second visualization is a bar graph showing the ratings for the top 10 highest budgeted movies from the data. The third visualization is a bar graph showing the ratings for the top 10 lowest budgeted movies from the data.

7.Documentation for each function that you wrote. This includes the input and output for each function (20 points)

<u>api.py functions:</u>

- get_titles():
  - Input = none
  - Output = returns list of movie titles
- get_request_url(list):
  - Input = list of movie titles
  - Output = returns a dictionary with movie title as the key and OMD API request url as the value
- get_ratings(dic):
  - Input = a dictionary with movie title as the key and OMD API request url as the value
  - Output = returns list of tuples with movie title and rating
- open_database(db_name):
  - Input = name of database file
  - Output = returns cur, conn
- create_id_table(cur, conn):
  - Input = cur, conn

○ Output = creates Movie_ids table in database
● add_id_data(data, cur, conn):
○ Input = list of tuples with movie title and rating, cur, conn
○ Output = adds movie id and movie title to Movie_ids table in database
● create_ratings_table(cur, conn):
○ Input = cur, conn
○ Output = creates Ratings table in database
● add_ratings_data(data, cur, conn):
○ Input = list of tuples with movie title and rating, cur, conn
○ Output = adds movie_id and rating to Ratings table in database

bs.py functions:
● get_titles():
○ Input = none
○ Output = returns list of movie titles
● get_budget(list):
○ Input = list of movie titles
○ Output = a list of tuples with (movie title, budget)
● open_database(name):
○ Input = name of database file
○ Output = returns cur, conn
● create_budget_table(cur, conn):
○ Input = cur, conn
○ Output = creates Budgets table in database
● add_budget_data(data, cur, conn):
○ Input = list of tuples with (movie title, budget), cur, conn
○ Output = adds movie_id and budget to Budgets table in database

calc.py functions:
● open_database(db_name):
○ Input = name of database file
○ Output = returns cur, conn
● under_30(cur):
○ Input = cur
○ Output = average rating for movies with a budget under $30 million
● between30_70(cur):
○ Input = cur
○ Output = average rating for movies with a budget between $30 million - $70 million
● between70_100(cur):
○ Input = cur

- Output = average rating for movies with a budget between $70 million - $100 million
- over_100(cur):
  - Input = cur
  - Output = average rating for movies with a budget over $100 million
- create_csv(file_name):
  - Input = file name of csv created in project folder
  - Output = creates csv file and adds average rating information to csv table

<u>visualizations.py functions:</u>

- create_visualization_1():
  - Input = None
  - Output = creates matplotlib bar chart with average rating for different budget groups
- create_visualization_2():
  - Input = None
  - Output = creates matplotlib bar chart with ratings for top 10 highest budgeted movies
- create_visualization_3():
  - Input = None
  - Output = creates matplotlib bar chart with ratings for top 10 lowest budgeted movies

8.You must also clearly document all resources you used. The documentation should be of the following form (20 points)

Date	Issue Description	Location of Resource	Result (did it solve the issue?)
December 4, 2022	Goodhousekeeping.com's article " The Best Movies of 2022 (So Far), and Our Most Anticipated Films for the Rest of the Year" provided a list of 100 movie titles.	www.goodhousekeeping.com/life/entertainment/g38502957/best-movies-2022/	Yes, we used BeautifulSoup with this resource that helped us compile a base list of 100 movie titles that we were able to use for our APIs.
December 4, 2022	The Open Movie Database API gave information such as IMDB ratings based on movie title.	https://www.omdbapi.com/	Yes, the Open Movie Database API helped us acquire an IMDB ratings for each title in our movie list.
December 8, 2022	Themoviedb.org	https://developers.the	Yes, the database was

	gives information about different movies like their title, budget, and release date when certain movie ids are called	moviedb.org/3/getting-started/search-and-query-for-details	able to give us the budgets for most of the movies on our list. However, there were some movies from our list that weren’t a part of the API database, so the resulting budget data for those movies is a 0.