

Final Project Report

Github Repository: https://github.com/final_project/

Team Name: I Wanna Go Home

Members: Chih-Hsiang Chang (43772622) & Xiwen Mark (06867643)

UPDATES AFTER PRESENTATION: Updated the scraping method so that it only inserts 25 titles at max

OMDB API KEY: “d4a57588”

Planned Goals and Data

The planned goal and purpose of this project is to evaluate the rating relationship between books and their movie adaptations using two APIs. The two APIs we are using for this project are Google Books and OMDB. We have checked the APIs and ensured that they contain sufficient data for our needs.

Actual Goals and Data

The actual goals and data continue to be built based on the planned goals and data. The overall objective remains the same, which is to identify the preferences and correlations between books and their movie adaptations. However, after execution, we added an additional website to scrape the necessary information. The website added during the actual execution is Goodreads, which contains a list of books that have been made into movies. Using the information from this website, we were able to match the books with their corresponding movies.

Problems and Challenges

Chih-Hsiang Chang:

My teammate and I had to go through several APIs before one combination finally worked. It took us much more time than expected. When I started working on the API and Web-Scraping, since its a new topic, it took a significant amount of time to start with. Every API has different parameters and we will have to know what works and what does not. After asking some questions regarding the tables and structures, we had to go back and fix several problems.

In the end, we used a different approach on the iterations, rather than looping through the titles we managed to just use title_id to request and present our data in the database instead of having both movie and book names in their respective tables.

Xiwen Mark:

We experienced confusion when designing the relational database structure because the data-gathering process contained repetitive information. Since we looped over the web-scraped data, this led to the same titles appearing across multiple tables. After consulting with the professor and TAs, we were able to identify a database structure that does not contain duplicate data.

For the calculation, we initially set a threshold requiring a sample size of at least 10. However, due to the number of null values in the book ratings, we removed the threshold and considered all observations that had any book rating data.

Calculations

Summary of calculations:

```
==== Analysis Summary ====  
  
Book vs Movie Preference (Question 1):  
    Total comparisons: 98  
    Books preferred: 85  
    Movies preferred: 12  
    Ties: 1  
  
Preference Percentages:  
    Books better %: 0.867  
    Movies better %: 0.122  
    Ties %: 0.010  
  
Correlation (Question 2):  
    Pearson r: 0.2963  
    p-value: 0.003054  
  
Regression (Question 2):  
    Slope: 0.1930  
    Intercept: 2.6587  
    r-value: 0.2963  
    p-value: 0.003054  
    std error: 0.063500
```

Question 1: Book vs. Movie Preference

We first counted the book and movie preferences, then calculated the preference percentages using the preference counts for the two categories and the total number of observations.

Question 2: Book vs. Movie Ratings Correlation

We computed the Pearson correlation coefficients using the SciPy library's Pearson method, along with the corresponding p-value for statistical significance. To better understand the strength of the correlation, we also performed a linear regression analysis, from which we obtained the slope and intercept.

Visualizations

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

Preference: Adapted Movies vs Original Books
Movies preferred

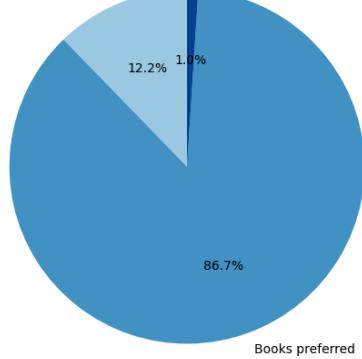


Figure 1: Preference: Adapted Movies vs Original Books Pie Chart

Preference: Adapted Movies vs Original Books

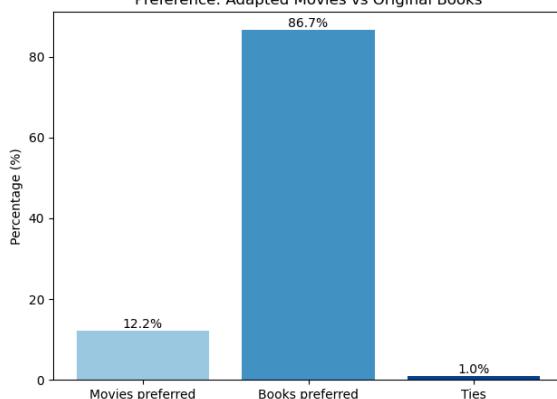


Figure 2: Preference: Adapted Movies vs Original Books Bar Chart

Interpretation:

There is around 86.7% of the data indicating a preference for books over movies, followed by 12.2% preferring movies and 1% indicating a tie. This suggests that the majority of people prefer the book over the movie in terms of book-to-movie adaptations.

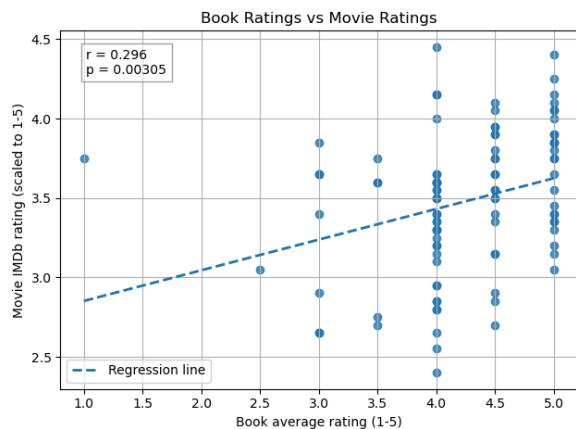


Figure 3: Book Ratings vs Movie Ratings Scatter Plot

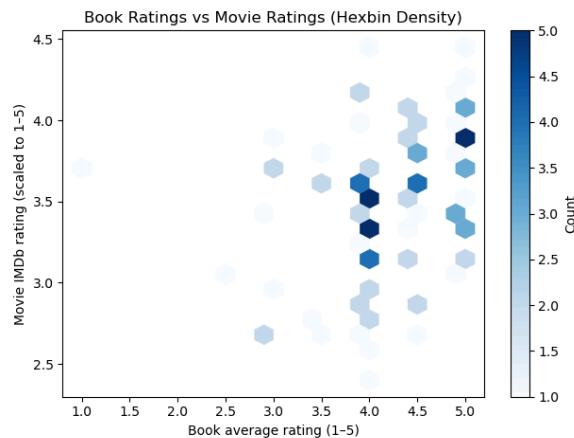


Figure 4: Book Ratings vs Movie Ratings Hexbin Density Plot

Interpretation:

The scatter plot shows a slightly upward trend; however, with a correlation coefficient of 0.296 and a p-value of 0.003, we can conclude that the correlation between the two variables is weak and statistically significant.

Using the hexagon bin density plot, we observe that most points cluster around book average ratings between 4.0 and 5.0, suggesting that higher-rated books tend to have more movie adaptations.

Additionally, movie ratings show more variation than book ratings. For book ratings between 4.0 and 5.0, movie ratings range from approximately 2.5 to 5.0, indicating that a higher book rating does not guarantee a higher movie adaptation rating.

There are only a few data points in the range where book ratings are 3.0 or below and movie ratings are above 3.5, which suggests that adaptations rarely outperform low-rated books. At the higher end, where most of the data cluster, around a book average rating of 4.5 and movie ratings around 3.5-3.7, the results suggest that books typically receive slightly higher ratings than their movie adaptations.

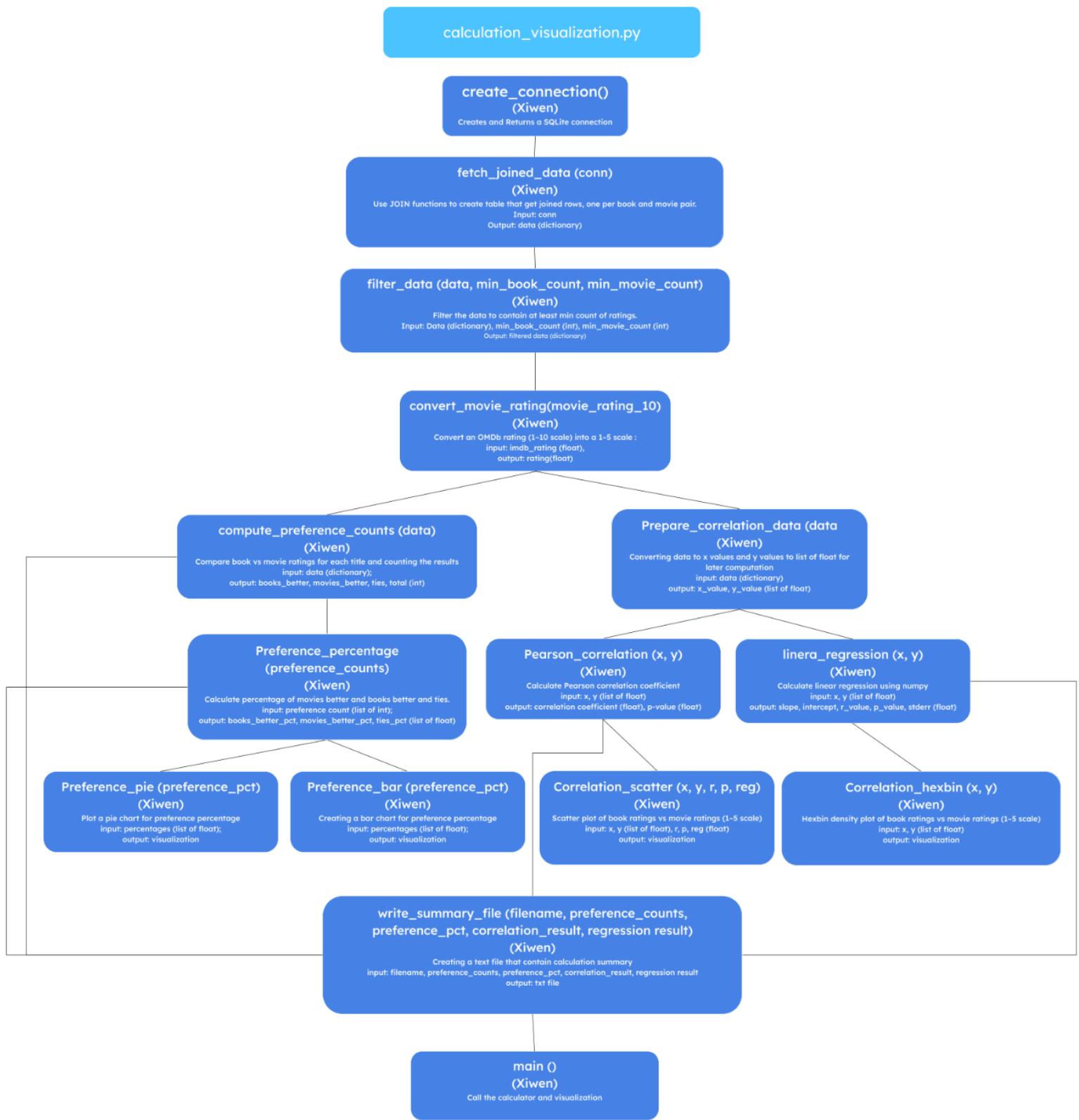
Instructions

- Check if OMDb API Key is there
- Run the `gather_data.py` 12 times
- Check if the database name in `calculation_visualization.py` the same as in `gather_data.py`
- Run the `calculation_visualization.py`
- Should have 4 graphs and 1 summary

To run the code, simply put your OMDb API Key into the “`OMDB_API_KEY`” in the **`gather_data.py`** file. If you want more data, simply go to the variable count “`maximumcount`” and change its values. Here, we limit it to 300 so we have sufficient data without making the output too abstract. Each run would scrape and append 25 data into the database, so one would have to run several tries before hitting the desired amount of data. After we have all sufficient data, go to **`calculation_visualization.py`** and run the code. It will generate 4 graphs (Pie Chart, Bar chart, Scatterplot, Hexbin Plot) and 1 output summary in the designated area. You will be able to see and analyze the data after running the code.

Updated Function Diagram





Resources Documentation

| Date | Issue Description | Location of Resource | Result |
|--------------------|--|---|--|
| Throughout Project | Brainstorming, Debug | ChatGPT | Successfully constructed structure and achieved project goals. |
| 11/23/2025 | Ensure the statistic concepts and the plot we are applying | https://mathematica.stackexchange.com/questions/28149/implementing-hexagon-binning-in-mathematica | Used hexagon plot for data visualization. |
| 11/23/2025 | Presentation template | Canva https://www.canva.com/templates/EAGqwpf_jeA/ | The presentation is in good structure and format. |