SI 206 Final Project Plan

a. What is your group's name?

Team name: TBnT

b. Who are the people in the group (first name, last name, umich email)?

Member 1: Ting Fong, Chen, tingfc@umich.edu

Member 2: Theo, Berry, theomb@umich.edu

c. What APIs/websites will you be gathering data from? The base URLs for the APIs/websites must be different for them to count as different APIs.

We will be gathering data from the APIs of MyAnimeList and MangaDex.

MyAnimeList base URL: https://api.myanimelist.net/v2

MangaDex base URL: https://api.mangadex.org

d. What data will you collect from each API/website and store in a database? Be specific.

We will be specifically comparing the popularity and ratings of anime and manga series from both APIs.

e. What data will you be calculating from the data in the database? Be specific.

We will be calculating aggregate review data from the most popular anime and manga series. We will look at the correlation between ratings (in float values) and popularity from the number of reviews (in integer values) for each show or book series, as well as to compare and contrast the correlations between anime and manga series. Our purpose is to investigate whether or not the popularity of an anime show or a manga book influences their rating, and how this relationship may differ between anime and manga.

f. What visualization package will you be using (Matplotlib, Plotly, Seaborn, etc)?

We will be using the seaborn package to create scatter and line plots.

g. What graphs/charts will you be creating?

Scatter plots and line plots that illustrate correlation between popularity and rating for both anime and manga series.

h. Who is responsible for what? Please note that all team members should do an equal amount of programming and total work.

Member 1: Finding raw data on the most popular anime and manga series and their ratings

Member 2: Creating graphs and charts that reflect correlations and relationships between popularity and ratings

Report

Project Goals

The APIs that we wanted to use were from the websites of MyAnimeList (MAL) and Mangadex.org. These two websites contained databases which had information on every anime and manga series available. Specifically, we wanted to collect data on the top 100 series in terms of popularity and compare them to their ratings, in order to determine the extent to which the popularity of a series influences its rating, as well as how popularity bias would compare between anime and manga series.

The metric for the popularity of a series would be determined by the number of online users that had indicated interest in the series or put the series in their personal watchlist. Meanwhile, the metric for the rating of a series would be determined by a mean float number out of 10.0 that would be averaged out across all the users who rated the show or book.

Ultimately, the purpose of this project is to visualize the correlation between popularity and rating for the top 100 anime and manga series. A scatter plot would be most suitable for illustrating this relationship, which could be enhanced by distinct color themes for anime and manga, as well as suitable scales and potential transformations to allow for more effective comparisons. Data visualization packages including Matplotlib, Seaborn and Pandas would be considered to assist with creating the graphs and for data analysis.

Actual Goals Achieved

Unfortunately, there were difficulties signing on to Mangadex, thereby we could not access the API stored in the website. As an alternative, we utilized the Anilist API instead, which also had data on manga series and even provided two sets of ID numbers: one that was uniquely for Anilist, the other being shared ID numbers with MAL, meaning that it would be easier to cross-reference and have two tables share primary integer keys.

The MAL API required an OAuth authentication token, which required me to first fill out an application to retrieve relevant information including client id, client secret, and code challenge, then write a set of functions that retrieved the token to get the API key. Meanwhile, the Anilist API was set up through a GraphQL query language system which required its own set of parameters when requesting the API key.

Eventually, we were able to access both APIs and gather data on the ID numbers, the popularity based on the number of users who rated the series, and the ratings themselves. A minor change we had was adjusting the scores on Anilist, which were out of 100 instead of 10, by simply dividing each score by 10, in order to match the scale of ratings on MAL.

Challenges

One major challenge we had was collaborating through GitHub. Firstly, even though the GitHub repository was made public, the owner forgot to give editing permissions to the partner, which took more time figuring out why he was not able to push the files he worked on onto the repo. Secondly, since we wanted to work on a single main document that had all the functions we wanted, including the API authentication and request functions, it was difficult updating the document from both sides. We learned the “git fetch” and “git pull” methods, which allowed the file to be synced up, but still required only one person to work on the file in order to prevent the person’s work from being overridden, which did initially happen several times before we decided to just have one person work on the code while the other works on the report. Lastly, there was an issue when the GitHub token expired for one member, which prevented them from committing the files. This was easily solved when they regenerated the token that would last them another 30 days, which was plenty of time to work on the code.

Another challenge came up when grabbing data on the popularity and ratings of anime series from the MAL API. The query for “anime rankings” only had anime series ranked in terms of popularity from 1 to 100, but we wanted to know the specific numbers of users who have rated each series from most to least, as well as the mean score for each series, so we could graph the popularity variable against the ratings variable from a more empirical perspective, not just with arbitrary ranking numbers. As a result, we had to access another query for “anime details” that contained actual information about the number of scoring users and the mean score. We first looped through the list of the top 100 series ranked by popularity within the json dictionary retrieved from “anime rankings”, grabbed the ID of each series, then used the ID as reference to get details on the number of scoring users and the mean score for each series. Finally, we input the ID, the number of users, and the mean score for all 100 series into a list of dictionaries that is nested within the json dictionary to be used later on for transferring information into the SQL database.

Calculations (with screenshots)


```
1  {"data": {"Page": {"pageInfo": {"total": 5000, "perPage": 50, "currentPage": 1, "lastPage": 100, "hasNextPage": true}, "media": [{"id": 30002, "idMal":
   2, "averageScore": 93, "popularity": 145413, "episodes": null}, {"id": 31706, "idMal": 1706, "averageScore": 92, "popularity": 64370, "episodes": null},
   {"id": 30656, "idMal": 656, "averageScore": 91, "popularity": 89321, "episodes": null}, {"id": 30013, "idMal": 13, "averageScore": 91, "popularity":
   139913, "episodes": null}, {"id": 114129, "idMal": 39486, "averageScore": 91, "popularity": 30773, "episodes": 1}, {"id": 30001, "idMal": 1,
   "averageScore": 90, "popularity": 64783, "episodes": null}, {"id": 20996, "idMal": 28977, "averageScore": 90, "popularity": 83851, "episodes": 51},
   {"id": 124194, "idMal": 42938, "averageScore": 90, "popularity": 110424, "episodes": 13}, {"id": 5114, "idMal": 5114, "averageScore": 90, "popularity":
   471749, "episodes": 64}, {"id": 125367, "idMal": 43608, "averageScore": 90, "popularity": 160979, "episodes": 13}, {"id": 104578, "idMal": 38524,
   "averageScore": 90, "popularity": 389286, "episodes": 10}, {"id": 150672, "idMal": 52034, "averageScore": 90, "popularity": 82782, "episodes": 11},
   {"id": 30642, "idMal": 642, "averageScore": 89, "popularity": 83842, "episodes": null}, {"id": 30051, "idMal": 51, "averageScore": 89, "popularity":
   45774, "episodes": null}, {"id": 151384, "idMal": 52198, "averageScore": 89, "popularity": 54664, "episodes": 4}, {"id": 11061, "idMal": 11061,
   "averageScore": 89, "popularity": 549049, "episodes": 148}, {"id": 116674, "idMal": 41467, "averageScore": 89, "popularity": 90214, "episodes": 13},
   {"id": 98478, "idMal": 35180, "averageScore": 89, "popularity": 98442, "episodes": 22}, {"id": 21745, "idMal": 35247, "averageScore": 89, "popularity":
   88063, "episodes": 7}, {"id": 9253, "idMal": 9253, "averageScore": 89, "popularity": 391184, "episodes": 24}, {"id": 30025, "idMal": 25, "averageScore":
   89, "popularity": 52766, "episodes": null}, {"id": 146984, "idMal": 51535, "averageScore": 89, "popularity": 91080, "episodes": 1}, {"id": 9969, "idMal":
   9969, "averageScore": 89, "popularity": 77469, "episodes": 51}, {"id": 15417, "idMal": 15417, "averageScore": 89, "popularity": 59840, "episodes": 13},
   {"id": 74489, "idMal": 44489, "averageScore": 89, "popularity": 43933, "episodes": null}, {"id": 98361, "idMal": 126479, "averageScore": 88,
   "popularity": 21879, "episodes": null}, {"id": 65243, "idMal": 35243, "averageScore": 88, "popularity": 73859, "episodes": null}, {"id": 64053, "idMal":
   34053, "averageScore": 88, "popularity": 5327, "episodes": null}, {"id": 46765, "idMal": 16765, "averageScore": 88, "popularity": 46679, "episodes":
   null}, {"id": 34632, "idMal": 4632, "averageScore": 88, "popularity": 114930, "episodes": null}, {"id": 31303, "idMal": 1303, "averageScore": 88,
   "popularity": 18955, "episodes": null}, {"id": 30657, "idMal": 657, "averageScore": 88, "popularity": 28862, "episodes": null}, {"id": 87395, "idMal":
   70345, "averageScore": 88, "popularity": 44912, "episodes": null}, {"id": 97889, "idMal": 34096, "averageScore": 88, "popularity": 58201, "episodes":
   12}, {"id": 820, "idMal": 820, "averageScore": 88, "popularity": 55469, "episodes": 110}, {"id": 103047, "idMal": 37987, "averageScore": 88,
   "popularity": 117113, "episodes": 1}, {"id": 164117, "idMal": 55016, "averageScore": 88, "popularity": 5219, "episodes": 1}, {"id": 135129, "idMal":
   null, "averageScore": 88, "popularity": 4575, "episodes": null}, {"id": 86635, "idMal": 90125, "averageScore": 88, "popularity": 84652, "episodes":
   null}, {"id": 30104, "idMal": 104, "averageScore": 88, "popularity": 39434, "episodes": null}, {"id": 143701, "idMal": null, "averageScore": 88,
   "popularity": 2683, "episodes": null}, {"id": 130003, "idMal": 47917, "averageScore": 88, "popularity": 95114, "episodes": 12}, {"id": 112275, "idMal":
   140765, "averageScore": 88, "popularity": 12563, "episodes": null}, {"id": 101338, "idMal": 37510, "averageScore": 88, "popularity": 293642, "episodes":
   13}, {"id": 30003, "idMal": 3, "averageScore": 88, "popularity": 61020, "episodes": null}, {"id": 20954, "idMal": 28851, "averageScore": 88,
   "popularity": 439904, "episodes": 1}, {"id": 19, "idMal": 19, "averageScore": 88, "popularity": 183448, "episodes": 74}, {"id": 17074, "idMal": 17074,
   "averageScore": 88, "popularity": 125275, "episodes": 26}, {"id": 30028, "idMal": 28, "averageScore": 88, "popularity": 31690, "episodes": null}, {"id":
   115166, "idMal": 123992, "averageScore": 87, "popularity": 9700, "episodes": null}]}}}
```

```
2  {"data": {"Page": {"pageInfo": {"total": 5000, "perPage": 50, "currentPage": 2, "lastPage": 100, "hasNextPage": true}, "media": [{"id": 94970, "idMal":
   89357, "averageScore": 87, "popularity": 28763, "episodes": null}, {"id": 2921, "idMal": 2921, "averageScore": 87, "popularity": 12436, "episodes": 47},
   {"id": 143838, "idMal": null, "averageScore": 87, "popularity": 1876, "episodes": null}, {"id": 15335, "idMal": 15335, "averageScore": 87, "popularity":
   38828, "episodes": 1}, {"id": 21, "idMal": 21, "averageScore": 87, "popularity": 420526, "episodes": null}, {"id": 140439, "idMal": 50172,
   "averageScore": 87, "popularity": 127508, "episodes": 12}, {"id": 30026, "idMal": 26, "averageScore": 87, "popularity": 67662, "episodes": null}, {"id":
   142329, "idMal": 47778, "averageScore": 87, "popularity": 243688, "episodes": 11}, {"id": 113024, "idMal": 40664, "averageScore": 87, "popularity":
   10035, "episodes": 1}, {"id": 85470, "idMal": 70261, "averageScore": 87, "popularity": 20861, "episodes": null}, {"id": 85135, "idMal": 56805,
   "averageScore": 87, "popularity": 65168, "episodes": null}, {"id": 53751, "idMal": 23751, "averageScore": 87, "popularity": 5957, "episodes": null},
   {"id": 31224, "idMal": 1224, "averageScore": 87, "popularity": 22386, "episodes": null}, {"id": 21698, "idMal": 32935, "averageScore": 87, "popularity":
   254829, "episodes": 10}, {"id": 21400, "idMal": 31758, "averageScore": 87, "popularity": 100047, "episodes": 1}, {"id": 4181, "idMal": 4181,
   "averageScore": 87, "popularity": 158903, "episodes": 24}, {"id": 110992, "idMal": 102875, "averageScore": 87, "popularity": 2318, "episodes": null},
   {"id": 101925, "idMal": 37491, "averageScore": 87, "popularity": 38203, "episodes": 14}, {"id": 101348, "idMal": 37521, "averageScore": 87, "popularity":
   294480, "episodes": 24}, {"id": 110277, "idMal": 40028, "averageScore": 87, "popularity": 409479, "episodes": 16}, {"id": 44893, "idMal": 14893,
   "averageScore": 87, "popularity": 14071, "episodes": null}, {"id": 121467, "idMal": 40434, "averageScore": 87, "popularity": 14232, "episodes": 12},
   {"id": 126403, "idMal": 44074, "averageScore": 87, "popularity": 72506, "episodes": 11}, {"id": 100805, "idMal": 119072, "averageScore": 87,
   "popularity": 23636, "episodes": null}, {"id": 98610, "idMal": 104039, "averageScore": 87, "popularity": 11549, "episodes": null}, {"id": 51525, "idMal":
   21525, "averageScore": 87, "popularity": 34754, "episodes": null}, {"id": 30336, "idMal": 336, "averageScore": 87, "popularity": 37053, "episodes":
   null}, {"id": 143726, "idMal": null, "averageScore": 87, "popularity": 1349, "episodes": null}, {"id": 75143, "idMal": 45143, "averageScore": 86,
   "popularity": 13804, "episodes": null}, {"id": 21733, "idMal": 33095, "averageScore": 86, "popularity": 41074, "episodes": 12}, {"id": 2904, "idMal":
   2904, "averageScore": 86, "popularity": 239409, "episodes": 25}, {"id": 131681, "idMal": 48583, "averageScore": 86, "popularity": 237026, "episodes":
   12}, {"id": 105778, "idMal": 116778, "averageScore": 86, "popularity": 208297, "episodes": null}, {"id": 85737, "idMal": 74697, "averageScore": 86,
   "popularity": 17976, "episodes": null}, {"id": 20751, "idMal": 24701, "averageScore": 86, "popularity": 41597, "episodes": 10}, {"id": 131586, "idMal":
   48569, "averageScore": 86, "popularity": 91899, "episodes": 12}, {"id": 37375, "idMal": 7375, "averageScore": 86, "popularity": 30136, "episodes": null},
   {"id": 113415, "idMal": 40748, "averageScore": 86, "popularity": 546889, "episodes": 24}, {"id": 106929, "idMal": 147171, "averageScore": 86,
   "popularity": 19153, "episodes": null}, {"id": 31133, "idMal": 1133, "averageScore": 86, "popularity": 55124, "episodes": null}, {"id": 30007, "idMal":
   7, "averageScore": 86, "popularity": 20510, "episodes": null}, {"id": 143841, "idMal": null, "averageScore": 86, "popularity": 1090, "episodes": null},
   {"id": 127720, "idMal": 45576, "averageScore": 86, "popularity": 160318, "episodes": 12}, {"id": 30044, "idMal": 44, "averageScore": 86, "popularity":
   21711, "episodes": null}, {"id": 111762, "idMal": 40417, "averageScore": 86, "popularity": 125976, "episodes": 25}, {"id": 107237, "idMal": 107931,
   "averageScore": 86, "popularity": 54108, "episodes": null}, {"id": 39115, "idMal": 9115, "averageScore": 86, "popularity": 13248, "episodes": null},
   {"id": 112641, "idMal": 40591, "averageScore": 86, "popularity": 289851, "episodes": 12}, {"id": 86559, "idMal": 85968, "averageScore": 86, "popularity":
   31958, "episodes": null}, {"id": 53900, "idMal": 23900, "averageScore": 86, "popularity": 4066, "episodes": null}]}}}
```

```
{"data": [{"id": 16498, "popularity": 3706044, "mean_score": 8.53, "num_episodes": 25}, {"id": 1535, "popularity": 3675465, "mean_score": 8.62,
"num_episodes": 37}, {"id": 5114, "popularity": 3138198, "mean_score": 9.1, "num_episodes": 64}, {"id": 30276, "popularity": 3026903, "mean_score": 8.5,
"num_episodes": 12}, {"id": 11757, "popularity": 2928393, "mean_score": 7.2, "num_episodes": 25}, {"id": 31964, "popularity": 2853988, "mean_score": 7.9,
"num_episodes": 13}, {"id": 38000, "popularity": 2760953, "mean_score": 8.51, "num_episodes": 26}, {"id": 20, "popularity": 2690223, "mean_score": 7.98,
"num_episodes": 220}, {"id": 22319, "popularity": 2675089, "mean_score": 7.79, "num_episodes": 12}, {"id": 11061, "popularity": 2623148, "mean_score":
9.04, "num_episodes": 148}, {"id": 32281, "popularity": 2566179, "mean_score": 8.85, "num_episodes": 1}, {"id": 25777, "popularity": 2537218,
"mean_score": 8.5, "num_episodes": 12}, {"id": 9253, "popularity": 2412850, "mean_score": 9.08, "num_episodes": 24}, {"id": 33486, "popularity": 2388207,
"mean_score": 8.12, "num_episodes": 25}, {"id": 1735, "popularity": 2322218, "mean_score": 8.26, "num_episodes": 500}, {"id": 19815, "popularity":
2287900, "mean_score": 8.08, "num_episodes": 12}, {"id": 40748, "popularity": 2208449, "mean_score": 8.64, "num_episodes": 24}, {"id": 35760,
"popularity": 2182738, "mean_score": 8.62, "num_episodes": 12}, {"id": 28851, "popularity": 2165412, "mean_score": 8.94, "num_episodes": 1}, {"id": 21,
"popularity": 2136743, "mean_score": 8.69, "num_episodes": 0}, {"id": 1575, "popularity": 2132283, "mean_score": 8.7, "num_episodes": 25}, {"id": 4224,
"popularity": 2104342, "mean_score": 8.08, "num_episodes": 25}, {"id": 23273, "popularity": 2090172, "mean_score": 8.65, "num_episodes": 22}, {"id":
38524, "popularity": 2071656, "mean_score": 9.06, "num_episodes": 10}, {"id": 31240, "popularity": 2071040, "mean_score": 8.24, "num_episodes": 25},
{"id": 36456, "popularity": 2064382, "mean_score": 8.05, "num_episodes": 25}, {"id": 20507, "popularity": 2062012, "mean_score": 7.95, "num_episodes":
12}, {"id": 6547, "popularity": 1997364, "mean_score": 8.06, "num_episodes": 13}, {"id": 22199, "popularity": 1979895, "mean_score": 7.47,
"num_episodes": 24}, {"id": 31043, "popularity": 1973437, "mean_score": 8.31, "num_episodes": 12}, {"id": 23755, "popularity": 1958013, "mean_score":
7.67, "num_episodes": 24}, {"id": 10620, "popularity": 1933257, "mean_score": 7.42, "num_episodes": 26}, {"id": 32182, "popularity": 1933075,
"mean_score": 8.49, "num_episodes": 12}, {"id": 24833, "popularity": 1894277, "mean_score": 8.09, "num_episodes": 22}, {"id": 21881, "popularity":
1887664, "mean_score": 6.7, "num_episodes": 24}, {"id": 20583, "popularity": 1850468, "mean_score": 8.44, "num_episodes": 25}, {"id": 37779,
"popularity": 1845110, "mean_score": 8.52, "num_episodes": 12}, {"id": 30831, "popularity": 1842264, "mean_score": 8.11, "num_episodes": 10}, {"id":
9919, "popularity": 1841767, "mean_score": 7.5, "num_episodes": 25}, {"id": 269, "popularity": 1835778, "mean_score": 7.91, "num_episodes": 366}, {"id":
40028, "popularity": 1817484, "mean_score": 8.8, "num_episodes": 16}, {"id": 22535, "popularity": 1764289, "mean_score": 8.34, "num_episodes": 24},
{"id": 1, "popularity": 1752520, "mean_score": 8.75, "num_episodes": 26}, {"id": 199, "popularity": 1741408, "mean_score": 8.78, "num_episodes": 1},
{"id": 30, "popularity": 1698043, "mean_score": 8.34, "num_episodes": 26}, {"id": 27899, "popularity": 1693753, "mean_score": 7.02, "num_episodes": 12},
{"id": 2904, "popularity": 1683236, "mean_score": 8.91, "num_episodes": 25}, {"id": 28223, "popularity": 1671898, "mean_score": 8.16, "num_episodes":
12}, {"id": 6702, "popularity": 1670226, "mean_score": 7.57, "num_episodes": 175}, {"id": 33352, "popularity": 1656390, "mean_score": 8.67,
"num_episodes": 13}, {"id": 18679, "popularity": 1645969, "mean_score": 8.04, "num_episodes": 24}, {"id": 37450, "popularity": 1617030, "mean_score":
8.25, "num_episodes": 13}, {"id": 38408, "popularity": 1612933, "mean_score": 7.91, "num_episodes": 25}, {"id": 37999, "popularity": 1609628,
"mean_score": 8.41, "num_episodes": 12}, {"id": 38691, "popularity": 1604383, "mean_score": 8.29, "num_episodes": 24}, {"id": 11111, "popularity":
1592360, "mean_score": 7.48, "num_episodes": 12}, {"id": 28171, "popularity": 1573100, "mean_score": 8.16, "num_episodes": 24}, {"id": 34134,
"popularity": 1570427, "mean_score": 7.5, "num_episodes": 12}, {"id": 13601, "popularity": 1555829, "mean_score": 8.34, "num_episodes": 22}, {"id": 3588,
"popularity": 1553414, "mean_score": 7.84, "num_episodes": 51}, {"id": 14719, "popularity": 1553197, "mean_score": 7.9, "num_episodes": 26}, {"id":
35849, "popularity": 1550093, "mean_score": 7.21, "num_episodes": 24}, {"id": 9989, "popularity": 1543146, "mean_score": 8.31, "num_episodes": 11},
{"id": 2001, "popularity": 1535758, "mean_score": 8.63, "num_episodes": 27}, {"id": 34572, "popularity": 1526240, "mean_score": 8.14, "num_episodes":
170}, {"id": 28999, "popularity": 1521570, "mean_score": 7.75, "num_episodes": 13}, {"id": 29803, "popularity": 1495078, "mean_score": 7.92,
"num_episodes": 13}, {"id": 15809, "popularity": 1492975, "mean_score": 7.75, "num_episodes": 13}, {"id": 35790, "popularity": 1481290, "mean_score":
7.98, "num_episodes": 25}, {"id": 226, "popularity": 1453262, "mean_score": 7.48, "num_episodes": 13}, {"id": 28121, "popularity": 1453016, "mean_score":
7.55, "num_episodes": 13}, {"id": 8074, "popularity": 1440335, "mean_score": 7.07, "num_episodes": 12}, {"id": 10087, "popularity": 1436834,
"mean_score": 8.28, "num_episodes": 13}, {"id": 32937, "popularity": 1422328, "mean_score": 8.27, "num_episodes": 10}, {"id": 121, "popularity": 1417648,
"mean_score": 8.11, "num_episodes": 51}, {"id": 37510, "popularity": 1412901, "mean_score": 8.8, "num_episodes": 13}, {"id": 34933, "popularity":
1394684, "mean_score": 7.25, "num_episodes": 12}, {"id": 30503, "popularity": 1391830, "mean_score": 8.16, "num_episodes": 13}, {"id": 40456,
"popularity": 1365463, "mean_score": 8.62, "num_episodes": 1}, {"id": 2167, "popularity": 1362945, "mean_score": 8, "num_episodes": 23}, {"id": 11617,
"popularity": 1359927, "mean_score": 7.34, "num_episodes": 12}, {"id": 14813, "popularity": 1358272, "mean_score": 8.01, "num_episodes": 13}, {"id":
6746, "popularity": 1352994, "mean_score": 8.11, "num_episodes": 24}, {"id": 28891, "popularity": 1346558, "mean_score": 8.63, "num_episodes": 25},
{"id": 5081, "popularity": 1346490, "mean_score": 8.33, "num_episodes": 15}, {"id": 30654, "popularity": 1345932, "mean_score": 8.49, "num_episodes":
25}, {"id": 37521, "popularity": 1321376, "mean_score": 8.73, "num_episodes": 24}, {"id": 37430, "popularity": 1318999, "mean_score": 8.14,
"num_episodes": 24}, {"id": 50265, "popularity": 1307345, "mean_score": 8.64, "num_episodes": 12}, {"id": 12189, "popularity": 1303603, "mean_score":
8.08, "num_episodes": 22}, {"id": 14741, "popularity": 1294958, "mean_score": 7.71, "num_episodes": 12}, {"id": 31478, "popularity": 1282276,
"mean_score": 7.82, "num_episodes": 12}, {"id": 34599, "popularity": 1275691, "mean_score": 8.66, "num_episodes": 13}, {"id": 26243, "popularity":
1263500, "mean_score": 7.49, "num_episodes": 12}, {"id": 40591, "popularity": 1238510, "mean_score": 8.64, "num_episodes": 12}, {"id": 431, "popularity":
1236972, "mean_score": 8.66, "num_episodes": 1}, {"id": 38671, "popularity": 1234126, "mean_score": 7.71, "num_episodes": 24}, {"id": 9756, "popularity":
1223967, "mean_score": 8.36, "num_episodes": 12}, {"id": 44511, "popularity": 1217057, "mean_score": 8.61, "num_episodes": 12}, {"id": 42897,
"popularity": 1215388, "mean_score": 8.2, "num_episodes": 13}]}
```

Visualizations (screenshots or image files)

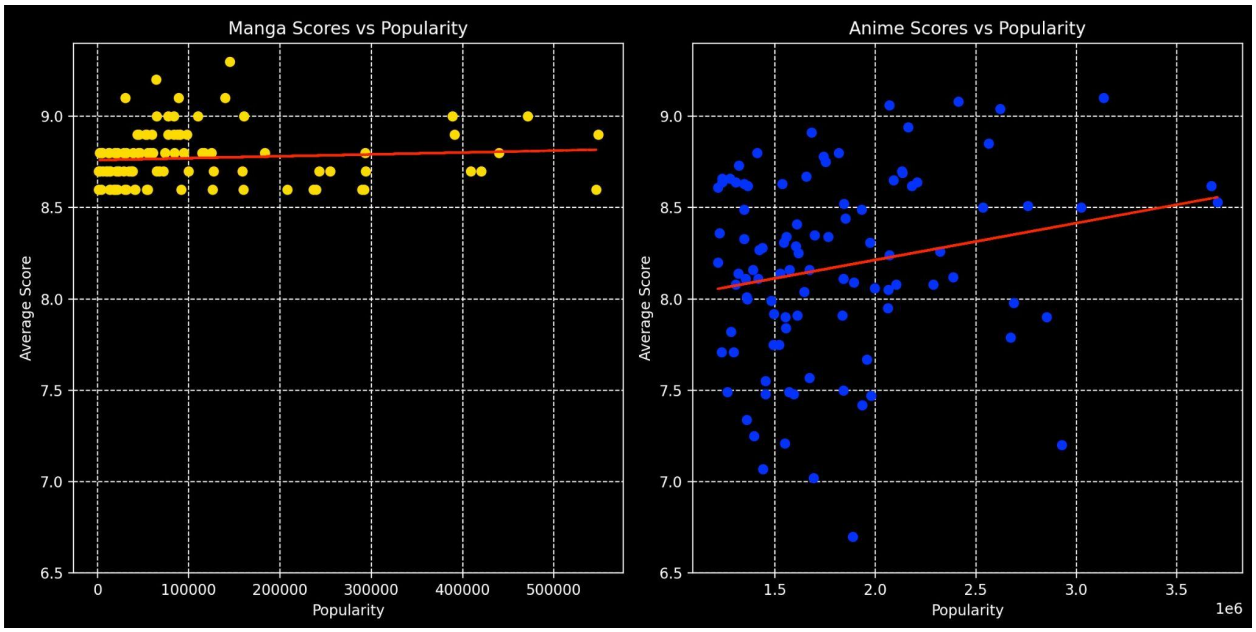

Figure 1: Manga (left) and Anime (right) Series' Scores vs Popularity

The two scatter plots shown above illustrate the relationship between the top 100 most popular manga and anime series and their ratings. On the x-axis is the popularity variable, which we have chosen to be the independent variable because we wanted to investigate if popularity is a factor that influences a series' rating. The popularity variable is measured by the number of users who have interacted with the series, whether they completed watching it, are currently watching it, have dropped it, or have planned to watch it in the future. The higher the number of users, the more popular a series is. On the y-axis is the rating variable, which is the dependent variable and is measured by the mean score calculated across all users' scores for each series, which also increases the accuracy of the rating by spreading out the distribution of scores. In addition, we have created two distinct color schemes to identify the different mediums: yellow for manga series, and blue for anime series. This way, readers will be able to easily identify each medium's level of popularity bias and compare them side-by-side.

Firstly, some comparisons can be made between the ratings of manga and anime series themselves. The left scatter plot indicates that the top 100 most popular manga series all have a rating above 8.5, while the right scatter plot indicates that a significant proportion of the most popular anime series have a rating below 8.5. This distributional difference suggests that the average audience views manga series as having higher quality than that of anime series. It may also be that due to anime series being more popular, as the number of anime users on the x-axis is one magnitude higher than that of manga users, there tends to be a wider range of opinions on the quality of the anime series, hence lowering the average score. Another reason could be that viewers who have seen both the anime adaptation and the original manga were

not satisfied with how accurately the adaptation depicted the elements of the original story, hence giving the anime a lower score than that of the manga.

Secondly, it is interesting to see how both anime and manga share a similar distribution across the popularity spectrum. As series become more popular, the ratings generally become less sparsely distributed and stay within a higher range. This phenomenon could feed into popularity bias as new viewers might be influenced by what others who have already seen the series have to say about it, making them give similar scores which leads to a highly concentrated cluster of scores for that particular series, resulting in a more stable average rating. Another reason might be that the most popular series also have very dedicated fanbases, meaning that some avid viewers might have created alternative accounts to maintain or bring up the rating for certain series, leading to a higher average score.

Lastly, when looking at the general trendline for both manga and anime, it is clear that popularity affects ratings to a much larger extent for anime than it does for manga. While manga series have a pretty flat trendline, indicating a very weak relationship between popularity and ratings, anime series have a steeper trendline, illustrating a positive correlation between popularity and ratings where the more popular a series becomes the higher the chance that its rating may increase. However, it is worthwhile noting that anime series also have very sparse distribution of data points that fall between ratings of 6.5 and 9.0 for the less popular series with a number of users below 2 million. This distribution implies that while some anime series may receive lower ratings due to being relatively less popular, there are other series that have similar numbers of users but receive much higher ratings. This revelation may suggest that while some anime may be quite niche and less well-known, they are cherished and praised by viewers who do know and have seen the series, which speaks to the quality of their production values. Another implication may be that some series are somewhat popular due to how notoriously bad their quality is, becoming more of a meme than a piece of work to be taken seriously, which can lead to a lower mean score. Nonetheless, it is visually evident that popularity bias plays an influential role in the way viewers perceive and rate anime series, more so than those who engage with manga series.

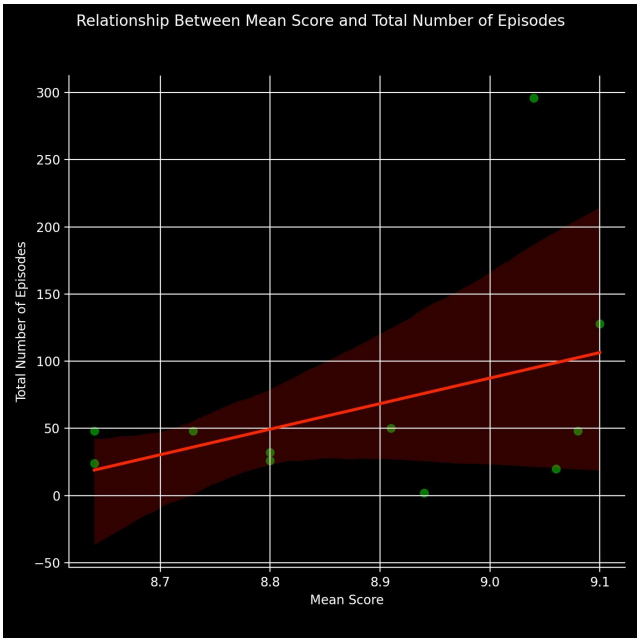

Figure 2: Relationship between Mean Score and Total Number of Episodes

The graph illustrates that the number of episodes does affect the mean score of anime adaptations of manga series and the original manga themselves. The red gradient in the back demonstrates the variance, which is important to have as there only a few data points available that match both the same anime and manga series.

Instructions for running the code

Step 1: Obtain the MAL API key

- Make sure you have a client ID and a client secret set up
- Import the secrets, json, and requests modules
- Run the get_new_code_verifier function, set the returned Code Verifier to a variable
- Run the print_new_authorisation_code function, use the Code Verifier as the input
- Copy and paste the generated url into a browser, then copy the authorization code after “?code=” which is given in the new url
- Run the generate_new_token function, using the authorization code and the Code Verifier as inputs
- If the token is successfully generated and saved, move on to the next step
- If the token has not been generated, go back to the previous steps and regenerate the token

Step 2: Gather relevant MAL data on anime series:

- Run the get_mal_ranking_data function, make sure the output is in the form of a dictionary so that it can be used as a json string for data manipulation later on

- This step may take a bit longer as the function needs to loop through 100 items in the API in order to generate a dictionary that contains all the relevant data for each anime series

Step 3: Gather relevant Anilist data on manga series:
- Run the anilist_pull function
- The function returns two sets of json data in increments of 50 items due to the API limit

Step 4: Storing collected data from both MAL and Anilist into combined database:
- Run the createBDfile function to input Anilist data into a table, use the two json dictionaries from step 3 as inputs for the function
- Run the add_mal_bd function to input MAL data into a table, use the json dictionary from step 2 as input for the function
- Run the add_data_join function to combine the two tables into a single database for visualizations in the next step

Step 5: Creating visualizations:
- Run the scatter_avg_popularity function to produce two different scatter plots where one illustrates the popularity bias of manga series and the other shows the popularity bias of anime series

Documentation for each function (including input & output for each function)

Getting the MAL API key:

Function 1: get_new_code_verifier()
Input: none.
Output: string containing the Code Verifier (length of string = 128 characters).

This function utilizes the secrets package by referencing the .token_urlsafe() function and generating a randomized string of 128 characters, which represents the Code Verifier.

The Code Verifier is used as a client's verification code through the PKCE protocol to prevent the OAuth workflow from becoming vulnerable to interception attacks. The Code Verifier is required for the API authentication process.

Function 2: print_new_authorisation_url(code_challenge):
Input: string containing the Code Verifier/Challenge generated from the previous function.

Output: print statement containing the url that leads the user to obtain the authorization code for the API application.

This function makes the client ID a global state so that it can create an authentication url which includes both the client ID and the Code Verifier. The user will need to click on the url, then copy the authorization code which is given after “?code=” in the newly generated url.

Function 3: generate_new_token(authorisation code, code_verifier):
Input: string containing the authorization code; string containing the Code Verifier.
Output: string containing the access token for obtaining the API key.

This function creates the url for accessing the API authentication token, which requires parameters including the client ID, the client secret, the authorization code, and the Code Verifier. Then it checks whether the request contains any potential errors. If not, then it generates a json file containing the access token, while printing a statement which lets the user know that the token has been generated and saved. The access token is necessary for generating the API key.

Function 4: print_user_info(access_token):
Input: string containing the access token.
Output: print statement confirming that the user now has access to the API database.

If the print statement is not generated and an error message appears, this means that the access token is not valid. The user will need to go back and run the previous functions to obtain another access token.

Function 5:
Input: none.
Output: output of Function 4.

This is the main function that runs all the previous functions to ensure that the API key can be successfully obtained.

Getting relevant MAL data:

Function 1: get_mal_ranking_data()
Input: none.
Output: json dictionary containing data on the ID of each anime series, the number of users that have rated the series (popularity), the mean score (rating), and the number of episodes.

This function first references the API query string for “Anime Rankings” with the parameters “ranking_type=bypopularity” and “limit=100”, in order to obtain data on the 100 most popular anime series in the form of a json dictionary. Then it loops through each anime series in the json dictionary, grabs the series ID, then references the API query string for “Anime Details”, which allows it to access the number of users, the mean score, and the episode number for each series. After that, the function stores all the data into a nested json dictionary and returns it. Within this json dictionary, the value of the key “data” is a list of dictionaries, where each dictionary contains relevant data for a particular anime series.

Getting relevant Anilist data:

Function 6: anilist_pull():
Input: none.
Output: json dictionary 1, json dictionary 2 containing relevant Anilist data for 100 items

Storing collected data into combined database:

Function 7: createBDfile(anilist_data1, anilist_data2):
Inputs: json dictionary 1, json dictionary 2 (from function 6)
Output: none. (data stored in database)

Function 8: add_MAL_bd(malJSON):
Input: json dictionary from function 5
Output: none. (data stored in database)

Function 9: add_data_join():
Input: none.
Output: none. (combining databases)

Creating visualizations:

Function 10: scatter_avg_popularity(db_file)
Input: the combined database
Output: visualizations (scatter plots) illustrating the popularity bias of manga and anime series

Documentation

Date	Description	Location of Resource	Result
4/19	Project Github	https://github.com/tfc0329/SI_206_final_project	Created project files including main code document, json file

			containing the API token, and the final database.
4/19	Authorisation flow for the new MAL API using OAuth 2.0	https://myanimelist.net/blog.php?eid=835707	Was able to learn the functions for obtaining the MAL API key.
4/19	MAL API References	https://myanimelist.net/apiconfig/references/api/v2#operation/anime_ranking_get	Used as reference for specific query strings that are necessary for obtaining relevant MAL data.
4/19	MAL Redirect URL Authorization Code	https://myanimelist.net/forum/?topicid=2061491	Found authorization code to obtain the MAL API key.
4/19	Formatting f strings	https://realpython.com/python-f-strings/	Learned to format strings without needing to use the .format function.
4/20	Anilist API	https://anilist.gitbook.io/anilist-apiv2-docs/overview/graphql/getting-started	How to set up GraphQL. Used https://anilist.co/graphiql? To help set it up
4/20	MatplotLib documentation	https://matplotlib.org/2.1.2/api/_as_gen/matplotlib.pyplot.grid.html	Helped me understand certain inputs
4/20	Seaborn documentation	https://seaborn.pydata.org/	Helped me understand certain inputs
4/21	ChatGPT	https://chat.openai.com/	Helped troubleshoot and correct code.