Table of Contents

Main Data Collection

December 10, 2024

[31]:
```
import requests
from bs4 import BeautifulSoup
import re
import pandas as pd
```

1 Title - Genre - Synopsis - Manga links

[33]:
```
page =1
elements_list = []
book_title = []
book_genre = []
book_synopsis = []
link_list = []
html_list = []


while page == 1:
    url = f'https://myanimelist.net/manga/genre/22/Romance?page={page}'
    response = requests.get(url)
    soup = BeautifulSoup(response.content)
    results = soup.find(id='myanimelist')
    elements_list = results.find_all('div', class_='seasonal-anime␣
 ↪js-seasonal-anime')
    html_list = results.find_all('h2', class_='h2_manga_title')


    # # Define a regular expression pattern to capture different parts
    # pattern = re.compile(r'(\D+), (\d+)(\D+)(\d+ vol, \d+ chp)')
    # Print the current page for debugging
    print(f"Fetching links from page {page}")


    for elements in elements_list:
        title = elements.find('h2', class_='h2_manga_title').text.strip()
        genre = elements.find('div', class_='genres-inner js-genre-inner').text.
 ↪strip()
        synopsis = elements.find('p', class_='preline').text.strip()
```

```
        genre = re.sub(r'\n+', ',', genre).strip()
        book_title.append(title)
        book_genre.append(genre)
        book_synopsis.append(synopsis)

    for elements in html_list:
        links = elements.find_all('a')
        for link in links:
            link_url = link['href']
            link_list.append(link_url)


    page = page + 1
```

```
Fetching links from page 1
```

2 Show same length of all lists

[35]:
```
# if links are not equal we wont be able to make a dataframe and if we did the␣
 ↪data might not match in each column
print(len(book_title))
print(len(book_genre))
print(len(book_synopsis))
print(len(link_list))
```

```
100
100
100
100
```

[37]:
```
link_list[13]
```

[37]:
```
'https://myanimelist.net/manga/25297/Prison_School'
```

3 Create a dataframe

[4]:
```
df = pd.DataFrame({'title': book_title,
                   'genre': book_genre,
                   'synopsis': book_synopsis,
                   'links': link_list,
})
df
```

[4]:
```
                                                title  \
0                                            Horimiya
1                                             Nisekoi
2                                          Ao Haru Ride
```

```
3                                                  Bakuman.
4                                       5-toubun no Hanayome
…                                                         …
17195  Outaishi-sama, Watashi Kondo koso Anata ni Kor…
17196  Sayonara Elevator: Kowakute Setsunai Gakkou no…
17197                              Shiina-san no Oshi Jijou
17198                            Soshite Megami wa Hohoemu
17199                          Yomei Ichinen, Otoko wo Kau

                               genre  \
0                             Romance
1                   Comedy,Romance
2                             Romance
3             Comedy,Drama,Romance
4      Award Winning,Comedy,Romance
…                               …
17195               Fantasy,Romance
17196    Drama,Romance,Supernatural
17197                Comedy,Romance
17198               Fantasy,Romance
17199                         Romance

                                                  synopsis  \
0      Although admired at school for her amiability …
1      When Raku Ichijou was young, he made a heartfe…
2      While most girls desire popularity among boys,…
3      Despite being a talented artist, middle school…
4      Considered a genius, high schooler Fuutarou Ue…
…                                                         …
17195  The poverty-stricken noblewoman Liesel was mur…
17196  1-3. Sayonara Elevator\r\n4. Kimi ga Kureta Ki…
17197                                  (No synopsis yet.)
17198                                  (No synopsis yet.)
17199                                  (No synopsis yet.)

                                                    links
0           https://myanimelist.net/manga/42451/Horimiya
1            https://myanimelist.net/manga/31499/Nisekoi
2       https://myanimelist.net/manga/24294/Ao_Haru_Ride
3             https://myanimelist.net/manga/9711/Bakuman
4      https://myanimelist.net/manga/103851/5-toubun_…
…                                                         …
17195  https://myanimelist.net/manga/162121/Outaishi-…
17196  https://myanimelist.net/manga/140291/Sayonara_…
17197  https://myanimelist.net/manga/165492/Shiina-sa…
17198  https://myanimelist.net/manga/151856/Soshite_M…
17199  https://myanimelist.net/manga/157563/Yomei_Ich…
```

```
[17200 rows x 4 columns]
```

[39]:
```
# df.to_csv('title-genre-synopsis-link.csv', index=False)

df = pd.read_csv('title-genre-synopsis-link.csv')
df.head()
```

[39]:

	title	genre	synopsis	links
0	Horimiya	Romance	Although admired at school for her amiability …	https://myanimelist.net/manga/42451/Horimiya
1	Nisekoi	Comedy,Romance	When Raku Ichijou was young, he made a heartfe…	https://myanimelist.net/manga/31499/Nisekoi
2	Ao Haru Ride	Romance	While most girls desire popularity among boys,…	https://myanimelist.net/manga/24294/Ao_Haru_Ride
3	Bakuman.	Comedy,Drama,Romance	Despite being a talented artist, middle school…	https://myanimelist.net/manga/9711/Bakuman
4	5-toubun no Hanayome	Award Winning,Comedy,Romance	Considered a genius, high schooler Fuutarou Ue…	https://myanimelist.net/manga/103851/5-toubun_…

4 Finds Date, Volumes, Chapters, Score, Rank, Themes, Manga Type

4.0.1 Here we are wanting to scrape the manga info from each manga page.

4.0.2 The things we are looking for are -

- Score - which tells us the average score of everyone who has submitted a score from 0-10
- Rank - based on score/favorites/popularity they are ranked from 1 to the amount of manga on the website
- Popularity - this metric is not shown how it is calculated
- Members - how many people have added it to there manga list
- Favorites - People can also add it to their favorites list and that is this number
- Genre - action, horror, drama, fantasy, adventure, comedy, and so on
- Theme - Time Travel, School, Workplace, samurai, historical, and so on
- volumes - number of volumes the manga has been around for
- chapters - total chapters all the manga collectivly contains
- publishing dates - the start date and end date if they exist
- status - shows if it is still creating new manga or not

[721]:
```
df = pd.read_csv('title-genre-synopsis-link.csv')
```

[56]:
```
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.chrome.service import Service as ChromeService
from selenium import webdriver
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import TimeoutException
options = Options()

options.headless = True
driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().
 ↪install()), options=options)
```

[725]:
```
import numpy as np
import pandas as pd
import time
import re


# Data structures - creates lists for all the names listed below
information, manga_type, manga_type_2, volume, chapter, status, date, theme,␣
 ↪score, rank, pop, favorite, links_2 = ([] for _ in range(13))
timer = 1

for link in df['links'][15000:17000]:
    print(timer)
    time.sleep(2)
    timer += 1
    theme_flag = False
    count = 0

    try:
        driver.set_page_load_timeout(300)  # Set to 300 seconds (5 minutes)

        driver.get(link)
        WebDriverWait(driver, 300).until(EC.presence_of_element_located((By.
 ↪CLASS_NAME, 'spaceit_pad')))
        # Statistics data extraction
        stats = driver.find_elements(By.CLASS_NAME, 'spaceit_pad')
        for stat in stats:
            if 'Type:' in stat.text:
                types = stat.text.strip('Type:').strip()
```

```
            links_2.append(link)
            if types == None:
                manga_type_2.append('Unknown')
            else:
                manga_type_2.append(types)

        if 'Volumes:' in stat.text:
            vol = stat.text.strip('Volumes:').strip()
            if vol == None:
                volume.append(np.nan)
            else:
                volume.append(vol)

        if 'Status:' in stat.text:
            finished = stat.text.strip('Status:').strip()
            if finished == None:
                status.append('Unknown')
            else:
                status.append(finished)


        if 'Chapters' in stat.text:
            chap = stat.text.strip('Chapters:').strip()
            if chap == None:
                chapter.append(np.nan)
            else:
                chapter.append(chap)

        if 'Published' in stat.text:
            pub = stat.text.strip('Published:').strip()
            if pub == None:
                date.append('Unknown')
            else:
                date.append(pub)


        if 'Score' in stat.text:
            scores = stat.text.strip('Score:').strip()
            if scores == None:
                score.append(np.nan)
            else:
                score.append(scores)

        if 'Ranked:' in stat.text:
            ranks = stat.text.strip('Ranked: #').strip()
            if ranks == None:
                rank.append(np.nan)
```

```
                else:
                    rank.append(ranks)


            if 'Popularity:' in stat.text:
                popularity = stat.text.strip('Popularity: #').strip()
                if popularity == None:
                    pop.append(np.nan)
                else:
                    pop.append(popularity)


            if 'Themes:' in stat.text or 'Theme:' in stat.text:
                themes = stat.text.replace('Themes:', '').replace('Theme:', '').
↪strip()

                print(stat.text)
                print(themes)
                if themes == '':
                    theme.append('Unknown')
                elif themes == None:
                    theme.append('Unknown')
                else:
                    theme.append(themes)
                    theme_flag = True
                    print(theme_flag)

            if 'Favorite' in stat.text:
                fav = stat.text.strip('Favorites:').strip()
                count += 1
                if fav == None:
                    favorite.append(np.nan)
                    print(fav)
                else:
                    favorite.append(fav)
                    print(fav)
                    print('-----')


        if not theme_flag:
            if count == 1:
                theme.append('Unknown')
                theme_flag = False
                print(theme_flag)
                print("------------------")
            else:
                continue
```

```
    except NoSuchElementException:
        print("Element not found")
        # Handle missing elements or data as needed
        continue
    except TimeoutException:
        print(f"Timeout while loading {link}")
        continue  # Skip to the next link if timeout occurs

# Don't forget to close the driver after the loop
driver.quit()
```

```
1
0
-----
False
------------------
2
0
-----
False
------------------
3
0
-----
False
------------------
4
0
-----
False
------------------
5
0
-----
False
------------------
6
0
-----
False
------------------
7
Theme: School
School
True
0
-----
```

```
0
-----
False
------------------
1997
Theme: School
School
True
0
-----
1998
0
-----
False
------------------
1999
0
-----
False
------------------
2000
0
-----
False
------------------
```

check the lengths of each list to make sure they match.

[715]:
```
# print(len(manga_type_2))
print(len(volume))
print(len(chapter))
print(len(theme))
print(len(favorite))
print(len(score))
print(len(rank))
print(len(pop))
print(len(status))
print(len(links_2))

print(theme[672])
favorite[671]
```

```
1543
1543
1543
1543
1543
1543
1543
```

```
1543
1543
Unknown
```

[715]:
```
'0'
```

[727]:
```
# Create a DataFrame from the list of dictionaries
df = pd.DataFrame({'type': manga_type_2,
                   'vol': volume,
                   'chap': chapter,
                   'status': status,
                   'published':date,
                   'theme':theme,
                   'favorites': favorite,
                   'score': score,
                   'rank': rank,
                   'popularity': pop,
                   'links': links_2})
# df.to_csv('stats_0-500.csv', index=False)
df
```

[727]:

	type	vol	chap	status	published	\
0	One-shot	Unknown	1	Finished	Jun 30, 2017	
1	Manga	1	6	Finished	Jan 15, 2016 to Sep 15, 2016	
2	Manga	Unknown	Unknown	Publishing	Nov 21, 2022 to ?	
3	Light Novel	1	12	Finished	Mar 2, 2020	
4	Light Novel	1	7	Finished	Dec 28, 2013	
…	…	…	…	…	…	
1984	Manga	Unknown	Unknown	Publishing	May 22, 2020 to ?	
1985	Manga	1	4	Finished	May 24, 2019 to Aug 24, 2019	
1986	One-shot	Unknown	1	Finished	Dec 23, 2021	
1987	Light Novel	1	8	Finished	Feb 12, 2016	
1988	Light Novel	1	Unknown	Finished	Jan 25, 2016	

	theme	favorites		score	rank	popularity	\
0	Unknown	0	N/A1	(scored by - users)	401482	50703	
1	Unknown	0	N/A1	(scored by - users)	503722	62485	
2	Unknown	0	N/A1	(scored by - users)	383802	60817	
3	Unknown	0	N/A1	(scored by - users)	490552	61084	
4	Unknown	0	N/A1	(scored by - users)	301902	62730	
…	…	…		…	…	…	
1984	Unknown	0	N/A1	(scored by - users)	385422	69859	
1985	School	0	N/A1	(scored by - users)	325732	68333	
1986	Unknown	0	N/A1	(scored by - users)	320172	69321	
1987	Unknown	0	N/A1	(scored by - users)	421532	69184	
1988	Unknown	0	N/A1	(scored by - users)	420622	69479	

	links

```
0     https://myanimelist.net/manga/161310/20-byou_Date
1     https://myanimelist.net/manga/135491/24H_Darli…
2     https://myanimelist.net/manga/153695/Ai_no_Clinic
3     https://myanimelist.net/manga/126564/Akunin_no…
4     https://myanimelist.net/manga/101641/Amai_Kuch…
…                                                   …
1984  https://myanimelist.net/manga/128827/V-idol_ni…
1985  https://myanimelist.net/manga/155523/Yorugata_…
1986  https://myanimelist.net/manga/152970/Yukine_Ko…
1987  https://myanimelist.net/manga/96648/Atsuki_Shi…
1988  https://myanimelist.net/manga/96360/Migawari_H…

[1989 rows x 11 columns]
```

[729]:
```
df.to_csv('stats_15000-17000.csv', index=False)
```

scrape-user-score-manga

December 10, 2024

[4]:
```
import pandas as pd
import re
from bs4 import BeautifulSoup
import requests
import time
```

[198]:
```
df = pd.read_csv('title-genre-synopsis-link.csv')
```

[200]:
```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service as ChromeService
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().
 ↪install()))
```

[12]:
```
user_list = []
```

1 Scrapes the saved links to get user names

2 Scrapes a specific managa link

[ ]:
```
information = []


timer = 1

for links in df['links'][500:1000]:
    print(links)
    print(timer)
    timer +=1
    num= 0sd
    # runs through each link and adds the show member page to get members links
    for i in range(1,10):

        url = f'{links}/stats?show={num}#members'
        response = requests.get(url)
        soup = BeautifulSoup(response.content)
```

```
        user_links = soup.find_all('table', class_="table-recently-updated")
        driver.get(url)

        num += 75

        time.sleep(5)
        driver.implicitly_wait(4)
        driver.set_page_load_timeout(300)  # Set to 300 seconds (5 minutes)

        for elements in user_links:
            # print(elements.prettify())
            base = elements.find_all('tr')
            for i in base:
                score = i.find('td', align='center',  class_='borderClass')


                if score and score.text.strip() != '-' and score.text.strip() !
 ↪= 'Score':

                    element_links = i.find_all('div', class_='di-tc va-m al␣
 ↪pl4')
                    for link in element_links:
                        user_names = link.find('a').text.strip()
                        user_list.append(user_names)
print(len(user_list))
```

3 Collect the users names

4 Creates a csv of the user list to use for future scraping

[ ]:
```
timer = 0
num = 0
for i in range(1,10):
        timer +=1
        print(timer)
        # url = f'{links}/stats?show={num}#members' # use to scrape each links␣
 ↪member list
        url = f'https://myanimelist.net/manga/118289/Skip_to_Loafer/stats?
 ↪show={num}#members'
        response = requests.get(url)
        soup = BeautifulSoup(response.content)
        user_links = soup.find_all('table', class_="table-recently-updated")
        driver.get(url)

        num += 75
```

```
        time.sleep(5)
        driver.implicitly_wait(4)
        driver.set_page_load_timeout(300)  # Set to 300 seconds (5 minutes)

        for elements in user_links:
            # print(elements.prettify())
            base = elements.find_all('tr')
            for i in base:
                score = i.find('td', align='center',  class_='borderClass')


                if score and score.text.strip() != '-' and score.text.strip() !
 ↪= 'Score':

                    element_links = i.find_all('div', class_='di-tc va-m al␣
 ↪pl4')
                    for link in element_links:
                        user_names = link.find('a').text.strip()
                        user_list.append(user_names)
print(len(user_list))
```

[206]:
```
series = pd.Series(user_list)
no_dup_user_list = series.unique()
print(len(no_dup_user_list))
user_names = pd.DataFrame({'users' :no_dup_user_list})
user_names.to_csv('list_of_users_set_4.csv', index=False)
```

```
60443
```

4.0.1 load selenium and start automated webdriver

[3]:
```
# Starts the automated webpage for scraping websites

from selenium import webdriver
from selenium.webdriver.chrome.service import Service as ChromeService
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().
 ↪install()))
```

4.0.2 If you previously scraped data and saved it use the code below to load it.

[12]:
```
# loads saved files
no_dup_user_list = pd.read_csv('list_of_users_set_4.csv')
no_dup_user_list.head()
```

[12]:
```
           users
0     Zichuwoss
1  Wham_Bam_Sam
2         Peepe
3        BissFo
4    Tolondrado
```

5 Scrapes the users manga list for manga names and scores

[289]:
```
from selenium.common.exceptions import StaleElementReferenceException,␣
 ↪NoSuchElementException, TimeoutException
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
import time
import re
import numpy as np

# List for storing the combined data
user_data = []
timer = 1

# Use the user_list above to add to the end of the web address
for users in no_dup_user_list['users'][20000:30000]:
    # Timer shows which iteration the loop is on or if it has failed
    print(timer)
    # Shows which user your scraping
    print(users)
    # Shows the users link
    print('link -', f'https://myanimelist.net/mangalist/{users}?
 ↪status=7&order=4&order2=0')
    # how much data have we scraped combined at each iteration
    print(len(user_data))

    timer += 1 # adds 1 per iteration to the timer

    try:
        # Provide the web address
        url = f'https://myanimelist.net/mangalist/{users}?
 ↪status=7&order=4&order2=0'

        # Use Selenium to open the URL
        driver.get(url)
        # Wait till the element is present
```

```
        WebDriverWait(driver, 50).until(EC.presence_of_element_located((By.
↪CSS_SELECTOR, 'td.data.title a.link')))
        # Allow some time for the page to load
        driver.implicitly_wait(5)
        # Redundency for how long we should wait if something is not present
        driver.set_page_load_timeout(50)

        # Find the title and score elements
        user_titles = driver.find_elements(By.CSS_SELECTOR, 'td.data.title a.
↪link') # looks for the titles
        user_scores = driver.find_elements(By.CSS_SELECTOR, 'td.data.score span.
↪score-label') # looks for the scores

        # Check if the lengths match
        if len(user_titles) == len(user_scores):
            # Iterate over titles and scores simultaneously
            for title_elem, score_elem in zip(user_titles, user_scores):
                title_text = title_elem.text
                score_text = score_elem.text
                user_data.append([users, title_text, score_text])
        else:
            print(f"Mismatch in lengths for user {users}:␣
↪Titles({len(user_titles)}) vs Scores({len(user_scores)})")
            # Handle the mismatch as needed (e.g., skip this user or handle␣
↪missing data)
            continue

    except NoSuchElementException:
        print("Element not found")
        # Handle missing elements or data as needed
        continue
    except TimeoutException:
        print(f"Timeout occurred for user {users}")
        continue  # Skip to the next link if timeout occurs
driver.quit()
```

```
1
jluong894
link - https://myanimelist.net/mangalist/jluong894?status=7&order=4&order2=0
0
2
TeaTheDemon
link - https://myanimelist.net/mangalist/TeaTheDemon?status=7&order=4&order2=0
300
3
HassaN__
link - https://myanimelist.net/mangalist/HassaN__?status=7&order=4&order2=0
```

```
link - https://myanimelist.net/mangalist/Bold_Pixie?status=7&order=4&order2=0
1343146
9996
ClareVoyance
link - https://myanimelist.net/mangalist/ClareVoyance?status=7&order=4&order2=0
1343153
9997
John_Wick123
link - https://myanimelist.net/mangalist/John_Wick123?status=7&order=4&order2=0
1343158
9998
Hanemiyakazutora
link -
https://myanimelist.net/mangalist/Hanemiyakazutora?status=7&order=4&order2=0
1343199
9999
Gindokusei
link - https://myanimelist.net/mangalist/Gindokusei?status=7&order=4&order2=0
1343210
10000
StyxParadise
link - https://myanimelist.net/mangalist/StyxParadise?status=7&order=4&order2=0
1343510
```

6 Seperate the Data into individual lists

[291]:
```
user = []
scores = []
titles = []

for i in user_data:
    user_name = i[0]
    user_score = i[2]
    user_title = i[1]
    user.append(user_name)
    scores.append(user_score)
    titles.append(user_title)
```

7 Create the dataframe

[293]:
```
# If the score and the titles length does match us the code below
df = pd.DataFrame({'user': user, 'score': scores, 'title': titles})
print(df)
```

```
                user score                                    title
0          jluong894    10                       Fullmetal Alchemist
1          jluong894    10                    Rascal Does Not Dream
```

```
2           jluong894    10                    Wotakoi: Love Is Hard for Otaku
3           jluong894    10         Rascal Does Not Dream of Bunny Girl Senpai
4           jluong894    10                      The Quintessential Quintuplets
…                 …   …                                                    …
1343552  StyxParadise     -  Douka Ore wo Houtteoite Kure: Nazeka Bocchi no…
1343553  StyxParadise     -                       Chigau Miyahara Omae ja Nai!
1343554  StyxParadise     -       The Reincarnated Marriage of a Hero and Sage
1343555  StyxParadise     -        Kusunoki's Flunking Her High School Glow-Up
1343556  StyxParadise     -                       Tune In to the Midnight Heart

[1343557 rows x 3 columns]
```

8 Export the data

[295]:
```
df3 = df
# convert the file to a csv file for further data analysis
df3.to_csv('user_rankings_v3_20000-30000.csv')
df3
```

[295]:
```
                user score                                                title
0           jluong894    10                                Fullmetal Alchemist
1           jluong894    10                             Rascal Does Not Dream
2           jluong894    10                    Wotakoi: Love Is Hard for Otaku
3           jluong894    10         Rascal Does Not Dream of Bunny Girl Senpai
4           jluong894    10                      The Quintessential Quintuplets
…                 …   …                                                    …
1343552  StyxParadise     -  Douka Ore wo Houtteoite Kure: Nazeka Bocchi no…
1343553  StyxParadise     -                       Chigau Miyahara Omae ja Nai!
1343554  StyxParadise     -       The Reincarnated Marriage of a Hero and Sage
1343555  StyxParadise     -        Kusunoki's Flunking Her High School Glow-Up
1343556  StyxParadise     -                       Tune In to the Midnight Heart

[1343557 rows x 3 columns]
```

[20]:
```
pd.read_csv('cleaned_user_data_v2.csv')
```

[20]:
```
         Unnamed: 0          user  score  \
0                 0       RakaanG    9.0
1                 1       RakaanG    9.0
2                 2       RakaanG    8.0
3                 3       RakaanG    8.0
4                 4       RakaanG    8.0
…               …           …    …
2238642     3346725  StyxParadise    9.0
2238643     3346726  StyxParadise    8.0
2238644     3346727  StyxParadise    8.0
2238645     3346728  StyxParadise    7.0
```

```
2238646     3346729  StyxParadise    5.0
```

```
                                          title
0                    My Hero Academia: Vigilantes
1        My Quiet Blacksmith Life in Another World
2                                My Hero Academia
3                                       Dr. Stone
4                                         Kujonin
…                                               …
2238642               I Had That Same Dream Again
2238643                      Pet Shop of Horrors
2238644            Lucifer and the Biscuit Hammer
2238645                        Chocolat no Mahou
2238646                Bonjour♪Koiaji Pâtisserie

[2238647 rows x 4 columns]
```

title genre link score newest

December 10, 2024

[141]:
```
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.chrome.service import Service as ChromeService
from selenium import webdriver
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import TimeoutException
import pandas as pd
import time

options = Options()

options.headless = True
driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().
 ↪install()), options=options)
```

1 Cold Start Data

[82]:
```
def scrape_all_manga_selenium(base_url, max_pages=20000):


    manga_titles = []
    manga_links = []
    page = 3000

    try:
        while True:
            if page >= max_pages:
                print(f"Reached maximum number of pages ({max_pages}). Stopping␣
 ↪Scraping.")
                break

            print(f"Scraping page: {page}")
            # Navigate to the current page
```

```
            driver.get(f"{base_url}?limit={page}")

            # Wait for the manga titles to load
            try:
                WebDriverWait(driver, 15).until(
                    EC.presence_of_element_located((By.CLASS_NAME,␣
↪'hoverinfo_trigger'))
                )
            except TimeoutException:
                print(f"Timeout on page {page}. Ending scraping.")
                page += 50
                continue

            # Find all manga title elements
            title_elements = driver.find_elements(By.CLASS_NAME,␣
↪'hoverinfo_trigger')

            # Break if no more manga entries found
            if not title_elements:
                print(f"No more pages to scrape. Ending on page {page}.")
                break

            # Extract title and link for each element
            for element in title_elements:
                title = element.text.strip()
                link = element.get_attribute('href')

                if not link.startswith("http"):
                    link = "https://myanimelist.net" + link

                manga_titles.append(title)
                manga_links.append(link)

            # Move to the next page
            page += 50
            time.sleep(2)  # Sleep to prevent rate limiting and give time to␣
↪observe

    finally:
        # Close the driver after scraping
        driver.quit()

    # Create a DataFrame with the collected manga titles and links
    manga_df = pd.DataFrame({
        'Title': manga_titles,
        'Link': manga_links
    })
```

```
    # Print the DataFrame to verify the results
    if manga_df.empty:
        print("No manga data found. Please check the page structure.")
    else:
        print(manga_df)

    # Optionally, save the DataFrame to a CSV file
    manga_df.to_csv('manga_list_1.csv', index=False)

# Replace 'https://myanimelist.net/manga' with the URL of the list page
scrape_all_manga_selenium('https://myanimelist.net/topmanga.php')
```

```
Scraping page: 3000
Scraping page: 3050
Scraping page: 3100
Scraping page: 3150
Scraping page: 3200
Scraping page: 3250
Scraping page: 3300
Scraping page: 3350
Scraping page: 3400
Scraping page: 3450
Scraping page: 3500
Scraping page: 3550
Scraping page: 3600
Scraping page: 3650
Scraping page: 3700
Scraping page: 3750
Scraping page: 3800
Scraping page: 3850
Scraping page: 3900
Scraping page: 3950
Scraping page: 4000
Scraping page: 4050
Scraping page: 4100
Scraping page: 4150
Scraping page: 4200
Scraping page: 4250
Scraping page: 4300
Scraping page: 4350
Scraping page: 4400
Scraping page: 4450
Scraping page: 4500
Scraping page: 4550
Scraping page: 4600
Scraping page: 4650
Scraping page: 4700
```

```
[33500 rows x 2 columns]
```

[125]:
```
df = pd.read_csv('manga_list_1.csv')
df.dropna(inplace=True)
df[df['Title'].str.contains('boku no hero', case=False)]
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 16750 entries, 1 to 33499
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Title   16750 non-null  object
 1   Link    16750 non-null  object
dtypes: object(2)
memory usage: 392.6+ KB
```

[117]:
```
df_2 = pd.read_csv('manga_list.csv')
df_2.dropna(inplace=True)
df_2.reset_index(drop=True)
```

[117]:

	Title	Link
0	Berserk	https://myanimelist.net/manga/2/Berserk
1	JoJo no Kimyou na Bouken Part 7: Steel Ball Run	https://myanimelist.net/manga/1706/JoJo_no_Kim…
2	Vagabond	https://myanimelist.net/manga/656/Vagabond
3	One Piece	https://myanimelist.net/manga/13/One_Piece
4	Monster	https://myanimelist.net/manga/1/Monster
…	…	…
3295	Otaku ni Yasashii Gal wa Inai!?	https://myanimelist.net/manga/144152/Otaku_ni_…
3296	Koukyuu no Karasu	https://myanimelist.net/manga/145695/Koukyuu_n…
3297	Yuutousei wa Unmei no Akaiito ni Sakaraitai	https://myanimelist.net/manga/146586/Yuutousei…
3298	After the Curtain Call	https://myanimelist.net/manga/147866/After_the…
3299	Kowareta Bokura no Senryaku Renai	https://myanimelist.net/manga/150374/Kowareta_…

```
[3300 rows x 2 columns]
```

[127]:
```
final_manga_info = pd.concat([df, df_2], ignore_index=True)
```

[440]:
```
final_manga_info['Title'].str.contains("My Hero")
```

[440]:
```
0        False
1        False
2        False
3        False
4        False
         …
20045    False
20046    False
20047    False
20048    False
20049    False
Name: Title, Length: 20050, dtype: bool
```

[129]:
```
final_manga_info.to_csv('title_link_extra.csv')
```

[ ]:
```

```

[145]:
```
import pandas as pd
from selenium import webdriver
from selenium.webdriver.chrome.service import Service as ChromeService
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.chrome.options import Options
from selenium.common.exceptions import TimeoutException, NoSuchElementException
from webdriver_manager.chrome import ChromeDriverManager
import time

def scrape_manga_details(manga_list_csv):
    # Load the CSV containing the manga links
    manga_df = pd.read_csv(manga_list_csv)


    # Set options to run Chrome in headless mode
    options = Options()
    options.headless = True

    # Initialize the WebDriver
    driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().
 ↪install()), options=options)
```

```
    # Lists to store manga details
    original_titles = []
    english_titles = []
    genres_list = []
    scores_list = []

    try:
        # Iterate over each manga link in the DataFrame
        for index, row in manga_df.iterrows():

            manga_link = row['Link'][0:1000]
            print(f"Scraping details for: {manga_link}")
            driver.get(manga_link)

            # Wait for the page to load
            try:
                WebDriverWait(driver, 10).until(
                    EC.presence_of_element_located((By.CLASS_NAME, 'h1-title'))
                )
            except TimeoutException:
                print(f"Timeout while accessing {manga_link}. Skipping...")
                original_titles.append(None)
                english_titles.append(None)
                genres_list.append(None)
                scores_list.append(None)
                continue

            # Scrape the original title
            try:
                original_title_element = driver.find_element(By.CLASS_NAME,␣
↪'h1-title')
                original_title = original_title_element.find_element(By.
↪TAG_NAME, 'span').text.strip()
            except NoSuchElementException:
                original_title = None

            # Scrape the English title if it exists
            try:
                english_title_element = original_title_element.find_element(By.
↪CLASS_NAME, 'title-english')
                english_title = english_title_element.text.strip()
            except NoSuchElementException:
                english_title = None

            # Scrape the genres
            try:
```

```
                genres_elements = driver.find_elements(By.XPATH, "//
↪span[@class='dark_text' and text()='Genres:']/following-sibling::a")
                genres = [genre.text.strip() for genre in genres_elements]
                genres_str = ", ".join(genres) if genres else None
            except NoSuchElementException:
                genres_str = None

            # Scrape the score
            WebDriverWait(driver, 300).until(EC.presence_of_element_located((By.
↪CLASS_NAME, 'spaceit_pad')))
        # Statistics data extraction
        stats = driver.find_elements(By.CLASS_NAME, 'spaceit_pad')
        for stat in stats:
            if 'Score' in stat.text:
                scores = stat.text.strip('Score:').strip()
                if scores == None:
                    score.append(np.nan)
                else:

                score = None

            # Append data to the lists
            original_titles.append(original_title)
            english_titles.append(english_title)
            genres_list.append(genres_str)
            scores_list.append(score)

            # Pause to prevent rate limiting
            time.sleep(1)

    finally:
        # Close the driver after scraping
        driver.quit()

    # Create new columns in the DataFrame with the collected details
    manga_df['Original Title'] = original_titles
    manga_df['English Title'] = english_titles
    manga_df['Genres'] = genres_list
    manga_df['Score'] = scores_list

    # Save the updated DataFrame to a new CSV file
    manga_df.to_csv('manga_details_with_scores_1.csv', index=False)
    print("Scraping completed. The updated details are saved to␣
↪'manga_details_with_scores.csv'.")

# Provide the path to your manga list CSV file and the maximum number of pages␣
↪to scrape
```

```
scrape_manga_details('title_link_extra.csv')
```

```
Scraping details for: https://myanimelist.net/manga/57185/Kyou_no_Cerberus
Scraping details for: https://myanimelist.net/manga/57723/Yoake_ni_Furu
Scraping details for:
https://myanimelist.net/manga/66645/Itsuka_Tomodachi_ja_Nakunaru_toshitemo
Scraping details for:
https://myanimelist.net/manga/69533/Non_Non_Biyori_Koushiki_Anthology
Scraping details for:
https://myanimelist.net/manga/79377/Yuusha_Gojo_Kumiai_Kouryuugata_Keijiban
Scraping details for:
https://myanimelist.net/manga/85081/Chikyuu_Umare_no_Anata_e
Scraping details for: https://myanimelist.net/manga/111512/Darling_in_the_FranXX
Scraping details for: https://myanimelist.net/manga/113294/Blue_Blue_Blue
Scraping details for: https://myanimelist.net/manga/115800/Tokyo_Ghoul_re__Quest
Scraping details for: https://myanimelist.net/manga/116702/Blue_Hearts
Scraping details for: https://myanimelist.net/manga/118143/Gohan_no_Otomo
Scraping details for:
https://myanimelist.net/manga/118862/Kanojo_ni_Naritai_Kimi_to_Boku
Scraping details for:
https://myanimelist.net/manga/123093/Fantasy_Bishoujo_Juniku_Ojisan_to
Scraping details for:
https://myanimelist.net/manga/124209/Real_mo_Tama_ni_wa_Uso_wo_Tsuku
Scraping details for:
https://myanimelist.net/manga/125806/Innsmouth_no_Kage__Lovecraft_Kessakushuu
Scraping details for: https://myanimelist.net/manga/128842/Minegishi-
san_wa_Ootsu-kun_ni_Tabesasetai
Scraping details for: https://myanimelist.net/manga/129280/Neko_to_Kiss
Scraping details for: https://myanimelist.net/manga/130039/Parasol_Doumei
Scraping details for: https://myanimelist.net/manga/131494/Kaishin_no_Ichigeki
Scraping details for: https://myanimelist.net/manga/132335/Death_Note_Tanpenshuu
Scraping details for: https://myanimelist.net/manga/18983/Fushigi_Yuugi_Gaiden
Scraping details for:
https://myanimelist.net/manga/25876/Hentai_Ouji_to_Warawanai_Neko
Scraping details for:
https://myanimelist.net/manga/26657/Tobaku_Haouden_Zero__Gyanki-hen
Scraping details for: https://myanimelist.net/manga/27765/Blood_and_Steel
Scraping details for: https://myanimelist.net/manga/34169/Nobunaga_Concerto
Scraping details for: https://myanimelist.net/manga/35907/JoJo_no_Kimyou_na_Bouk
en_II__Golden_Heart_Golden_Ring
Scraping details for: https://myanimelist.net/manga/39959/Senpai
Scraping details for:
https://myanimelist.net/manga/41539/Mahouka_Koukou_no_Yuutousei
Scraping details for:
https://myanimelist.net/manga/42069/Le_Th%C3%A9%C3%A2tre_de_A
Scraping details for:
https://myanimelist.net/manga/89985/Shitayomi_Danshi_to_Toukou_Joshi
Scraping details for:
```

```
Scraping details for: https://myanimelist.net/manga/105875/Okuru_Kotoba
Scraping details for: https://myanimelist.net/manga/107500/Munou_na_Nana
Scraping details for: https://myanimelist.net/manga/108034/Rougo_ni_Sonaete_Isek
ai_de_8-manmai_no_Kinka_wo_Tamemasu
Scraping details for: https://myanimelist.net/manga/175692/Belladonna_no_Koibito
Scraping details for: https://myanimelist.net/manga/18197/Musunde_Hiraite
Scraping details for: https://myanimelist.net/manga/20855/A-Channel
Scraping details for: https://myanimelist.net/manga/22870/Six_Half
Scraping details for: https://myanimelist.net/manga/23384/Tokyo_Rock_Shounen
Scraping details for: https://myanimelist.net/manga/23463/Otome_no_Teikoku
Scraping details for:
https://myanimelist.net/manga/24879/Sugar_Dark__Umerareta_Yami_to_Shoujo
Scraping details for: https://myanimelist.net/manga/31185/14-sai_no_Koi
Scraping details for:
https://myanimelist.net/manga/33511/Papa_to_Mama_Hajimemashita
Scraping details for:
https://myanimelist.net/manga/36717/Joujuu_Senjin_Mushibugyou
Scraping details for:
https://myanimelist.net/manga/37405/Sword_Art_Online__Material_Edition
Scraping details for: https://myanimelist.net/manga/43789/Houkago_x_Ponytail
Scraping details for:
https://myanimelist.net/manga/47585/Final_Fantasy_VII__On_the_Way_to_a_Smile
Scraping details for: https://myanimelist.net/manga/50587/Fukumenkei_Noise
Scraping details for: https://myanimelist.net/manga/138443/1_Second
Scraping details for:
https://myanimelist.net/manga/143287/Kedamono_Arashi__Hold_Me_Baby
Scraping details for: https://myanimelist.net/manga/143884/Ako_to_Bambi
Scraping details for:
https://myanimelist.net/manga/144152/Otaku_ni_Yasashii_Gal_wa_Inai
Scraping details for: https://myanimelist.net/manga/145695/Koukyuu_no_Karasu
Scraping details for:
https://myanimelist.net/manga/146586/Yuutousei_wa_Unmei_no_Akaiito_ni_Sakaraitai
Scraping details for:
https://myanimelist.net/manga/147866/After_the_Curtain_Call
Scraping details for:
https://myanimelist.net/manga/150374/Kowareta_Bokura_no_Senryaku_Renai
Scraping completed. The updated details are saved to
'manga_details_with_scores.csv'.
```

[242]:
```
test = pd.read_csv('manga_details_with_scores_1.csv')
test
```

[242]:

	Title	\
0	Kyou no Cerberus	
1	Yoake ni Furu,	
2	Itsuka Tomodachi ja Nakunaru toshitemo	
3	Non Non Biyori Koushiki Anthology	

```
4          Yuusha Gojo Kumiai Kouryuugata Keijiban
…                                                 …
20045              Otaku ni Yasashii Gal wa Inai!?
20046                           Koukyuu no Karasu
20047  Yuutousei wa Unmei no Akaiito ni Sakaraitai
20048                       After the Curtain Call
20049           Kowareta Bokura no Senryaku Renai

                                                      Link  \
0      https://myanimelist.net/manga/57185/Kyou_no_Ce…
1      https://myanimelist.net/manga/57723/Yoake_ni_Furu
2      https://myanimelist.net/manga/66645/Itsuka_Tom…
3      https://myanimelist.net/manga/69533/Non_Non_Bi…
4      https://myanimelist.net/manga/79377/Yuusha_Goj…
…                                                     …
20045  https://myanimelist.net/manga/144152/Otaku_ni_…
20046  https://myanimelist.net/manga/145695/Koukyuu_n…
20047  https://myanimelist.net/manga/146586/Yuutousei…
20048  https://myanimelist.net/manga/147866/After_the…
20049  https://myanimelist.net/manga/150374/Kowareta_…

                                          Original Title  \
0                   Kyou no Cerberus\nToday's Cerberus
1                                          Yoake ni Furu,
2      Itsuka Tomodachi ja Nakunaru toshitemo\nSomeda…
3                    Non Non Biyori Koushiki Anthology
4                Yuusha Gojo Kumiai Kouryuugata Keijiban
…                                                     …
20045  Otaku ni Yasashii Gal wa Inai!?\nGal Can't be …
20046       Koukyuu no Karasu\nRaven of the Inner Palace
20047  Yuutousei wa Unmei no Akaiito ni Sakaraitai\nT…
20048                              After the Curtain Call
20049                  Kowareta Bokura no Senryaku Renai

                                            English Title  \
0                                        Today's Cerberus
1                                                     NaN
2                    Someday a friend will be a stranger
3                                                     NaN
4                                                     NaN
…                                                     …
20045                       Gal Can't be Kind to Otaku!?
20046                          Raven of the Inner Palace
20047  The Honor Student Wants to Go Against the Red …
20048                                                 NaN
20049                                                 NaN
```

```
                                Genres  Score
0      Comedy, Romance, Supernatural    NaN
1                                NaN    NaN
2                                NaN    NaN
3              Comedy, Slice of Life    NaN
4         Adventure, Comedy, Fantasy    NaN
…                                …    …
20045                Comedy, Romance    NaN
20046                            NaN    NaN
20047                            NaN    NaN
20048              Drama, Girls Love    NaN
20049                            NaN    NaN

[20050 rows x 6 columns]
```

[176]:
```
test['Score'].value_counts
```

[176]:
```
<bound method IndexOpsMixin.value_counts of 0       NaN
1       NaN
2       NaN
3       NaN
4       NaN
         ..
20045   NaN
20046   NaN
20047   NaN
20048   NaN
20049   NaN
Name: Score, Length: 20050, dtype: float64>
```

[434]:
```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import TimeoutException
from selenium.webdriver.common.action_chains import ActionChains
import pandas as pd
import time

def scrape_all_manga_selenium(base_url, max_pages=5000):
    # Set options to run Chrome in headless mode
    options = Options()
    options.headless = True

    # Initialize the WebDriver
    driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().
 ↪install()), options=options)
```

```
    manga_titles = []
    manga_links = []
    manga_scores = []
    manga_image_links = []
    manga_genres = []

    page = 0

    try:
        while True:
            if page >= max_pages:
                print(f"Reached maximum number of pages ({max_pages}). Stopping␣
↪Scraping.")
                break

            print(f"Scraping page: {page}")
            # Navigate to the current page
            driver.get(f"{base_url}?limit={page}")

            # Wait for the manga entries to load
            try:
                WebDriverWait(driver, 5).until(
                    EC.presence_of_element_located((By.CLASS_NAME,␣
↪'hoverinfo_trigger'))
                )
            except TimeoutException:
                print(f"Timeout on page {page}. Ending scraping.")
                page += 50
                continue

            # Find all manga entries in the current page
            manga_rows = driver.find_elements(By.CLASS_NAME, 'ranking-list')
            manga_detail = driver.find_elements(By.CLASS_NAME, 'hoverinfo right␣
↪top')
            for detail in manga_detail:
                print(detail.text)

            # Break if no more manga entries are found
            if not manga_rows:
                print(f"No more pages to scrape. Ending on page {page}.")
                break

            # Extract data for each manga entry
            for row in manga_rows:

                try:
```

```
                    # Title and link
                    title_element = row.find_element(By.CLASS_NAME,␣
↪'hoverinfo_trigger')
                    title = title_element.text.strip()
                    link = title_element.get_attribute('href')
                    info = title_element.get_attribute('rel')
                    info_cleaned = info.lstrip('#')  # Remove the leading '#'␣
↪character
                    # Scroll the element into view using JavaScript
                    driver.execute_script("arguments[0].scrollIntoView(true);",␣
↪title_element)
                    time.sleep(0.5)  # Give time for the scroll to settle

                    # Ensure link starts with "http"
                    if not link.startswith("http"):
                        link = "https://myanimelist.net" + link

                    # Score
                    try:
                        score_element = row.find_element(By.CLASS_NAME, 'score')
                        score = score_element.text.strip()
                    except Exception:
                        score = "N/A"

                    # Image link
                    try:
                        image_element = row.find_element(By.TAG_NAME, 'img')
                        image_link = image_element.get_attribute('data-src')
                    except Exception:
                        image_link = "N/A"

                     # Hover over the title to reveal the bubble
                    actions = ActionChains(driver)
                    actions.move_to_element(title_element).perform()



                    # Wait for the hover bubble to appear
                    try:
                        hover_bubble = WebDriverWait(driver, 1.5).until(
                            EC.visibility_of_element_located((By.CLASS_NAME,␣
↪'hoverinfo'))
                        )

                        hover_text = hover_bubble.text
```

```
                        # Extract genres from the hover text
                        if "Genres:" in hover_text:
                            genres_start = hover_text.find("Genres:") +␣
↪len("Genres:")
                            genres_end = hover_text.find("\n", genres_start)
                            genres = hover_text[genres_start:genres_end].strip()
                        else:
                            genres = None

                    except TimeoutException:
                        genres = None

                    try:
                        info_bubble = WebDriverWait(driver, .1).until(
                            EC.visibility_of_element_located((By.ID,␣
↪info_cleaned))
                        )
                        # print(info_bubble.text)
                        for i in info_bubble.text.splitlines():
                            if i.startswith("Genres:"):
                                genres = i.replace("Genres:", "").strip()
                                print(genres)

                    except Exception:
                        continue



                    # Append data to lists
                    manga_titles.append(title)
                    manga_links.append(link)
                    manga_scores.append(score)
                    manga_image_links.append(image_link)
                    manga_genres.append(genres)

                    # Reset hover
                    actions.move_by_offset(10, 10).perform()




                except Exception as e:
                    print(f"Error scraping row: {e}")
                    continue

            # Move to the next page
```

```
            page += 50


    finally:
        # Close the driver after scraping
        driver.quit()

    # Create a DataFrame with the collected data
    manga_df = pd.DataFrame({
        'Title': manga_titles,
        'Link': manga_links,
        'Score': manga_scores,
        'Image Link': manga_image_links,
        'Genres': manga_genres
    })

    # # Print the DataFrame to verify the results
    # if manga_df.empty:
    #     print("No manga data found. Please check the page structure.")
    # else:
    #     print(manga_df)

    # Optionally, save the DataFrame to a CSV file
    manga_df.to_csv('manga_list_3.csv', index=False)
    print("Data saved to manga_list_3.csv")
# Replace 'https://myanimelist.net/topmanga.php' with the URL of the list page
scrape_all_manga_selenium('https://myanimelist.net/topmanga.php')
```

```
Scraping page: 0
Action, Adventure, Mystery, Supernatural
Action, Adventure, Award Winning
Action, Adventure, Fantasy
Award Winning, Drama, Mystery
Award Winning, Sports
Action, Adventure, Boys Love, Supernatural
Action, Adventure, Award Winning, Drama, Fantasy
Action, Award Winning
Action, Adventure, Fantasy
Action, Mystery, Supernatural, Suspense
Drama, Slice of Life
Action, Drama, Fantasy
Award Winning, Drama, Sports
Award Winning, Drama, Mystery, Sci-Fi
Drama, Slice of Life, Sports
Action, Comedy, Mystery, Romance, Supernatural
Action, Adventure, Boys Love, Mystery, Supernatural
Award Winning, Comedy, Slice of Life
Award Winning, Comedy
```

```
Sports, Ecchi
Drama, Fantasy, Horror
Award Winning, Comedy, Fantasy, Slice of Life, Supernatural
Award Winning, Romance
Drama, Sports
Action, Comedy, Fantasy, Romance, Ecchi
Drama, Supernatural
Comedy, Supernatural
Action, Fantasy
Fantasy, Girls Love
Action, Sci-Fi
Comedy, Romance, Supernatural
Comedy, Drama, Mystery
Boys Love, Slice of Life
Action, Fantasy
Comedy, Romance
Gourmet, Slice of Life
Drama, Fantasy, Romance
Boys Love, Comedy
Fantasy, Romance, Supernatural
Action, Adventure, Drama, Fantasy, Horror
Drama, Romance
Action, Adventure, Ecchi
Comedy, Drama, Romance, Supernatural
Comedy, Drama, Romance
Award Winning, Sports
Comedy, Romance
Drama, Romance
Horror, Mystery, Romance, Supernatural
Comedy, Drama, Romance
Action, Adventure, Fantasy
Comedy, Fantasy
Reached maximum number of pages (5000). Stopping Scraping.
Data saved to manga_list_3.csv
```

[444]:
```
df_5 = pd.read_csv('manga_list_3.csv')
df_5.isna().sum()
```

[444]:
```
Title         4997
Link             0
Score            0
Image Link       0
Genres        1254
dtype: int64
```

[208]:
```
df_2 = pd.read_csv('manga_image_list.csv')
```

[222]:
```
merged_data = pd.merge(test, df_2[['Link', 'Score', 'Image Link']] , on='Link',␣
↪how='left')
merged_data.drop('Score_x', axis=1, inplace=True)
merged_data.rename(columns={'Score_y': 'Score'})
```

[222]:

```
                                          Title  \
0                               Kyou no Cerberus
1                                 Yoake ni Furu,
2          Itsuka Tomodachi ja Nakunaru toshitemo
3               Non Non Biyori Koushiki Anthology
4         Yuusha Gojo Kumiai Kouryuugata Keijiban
…                                             …
20045             Otaku ni Yasashii Gal wa Inai!?
20046                           Koukyuu no Karasu
20047  Yuutousei wa Unmei no Akaiito ni Sakaraitai
20048                      After the Curtain Call
20049           Kowareta Bokura no Senryaku Renai

                                                   Link  \
0      https://myanimelist.net/manga/57185/Kyou_no_Ce…
1      https://myanimelist.net/manga/57723/Yoake_ni_Furu
2      https://myanimelist.net/manga/66645/Itsuka_Tom…
3      https://myanimelist.net/manga/69533/Non_Non_Bi…
4      https://myanimelist.net/manga/79377/Yuusha_Goj…
…                                                    …
20045  https://myanimelist.net/manga/144152/Otaku_ni_…
20046  https://myanimelist.net/manga/145695/Koukyuu_n…
20047  https://myanimelist.net/manga/146586/Yuutousei…
20048  https://myanimelist.net/manga/147866/After_the…
20049  https://myanimelist.net/manga/150374/Kowareta_…

                                       Original Title  \
0                  Kyou no Cerberus\nToday's Cerberus
1                                       Yoake ni Furu,
2      Itsuka Tomodachi ja Nakunaru toshitemo\nSomeda…
3                    Non Non Biyori Koushiki Anthology
4              Yuusha Gojo Kumiai Kouryuugata Keijiban
…                                                    …
20045  Otaku ni Yasashii Gal wa Inai!?\nGal Can't be …
20046      Koukyuu no Karasu\nRaven of the Inner Palace
20047  Yuutousei wa Unmei no Akaiito ni Sakaraitai\nT…
20048                           After the Curtain Call
20049                Kowareta Bokura no Senryaku Renai

                                      English Title  \
0                                   Today's Cerberus
1                                               NaN
```

```
2                    Someday a friend will be a stranger
3                                                    NaN
4                                                    NaN
…                                                      …
20045                        Gal Can't be Kind to Otaku!?
20046                           Raven of the Inner Palace
20047  The Honor Student Wants to Go Against the Red …
20048                                                NaN
20049                                                NaN

                              Genres  Score  \
0      Comedy, Romance, Supernatural   7.49
1                                NaN   7.49
2                                NaN   7.49
3              Comedy, Slice of Life   7.49
4         Adventure, Comedy, Fantasy   7.49
…                                …    …
20045               Comedy, Romance   7.45
20046                            NaN   7.47
20047                            NaN   7.45
20048             Drama, Girls Love   7.45
20049                            NaN   7.45

                                                Image Link
0      https://cdn.myanimelist.net/r/50x70/images/man…
1      https://cdn.myanimelist.net/r/50x70/images/man…
2      https://cdn.myanimelist.net/r/50x70/images/man…
3      https://cdn.myanimelist.net/r/50x70/images/man…
4      https://cdn.myanimelist.net/r/50x70/images/man…
…                                                    …
20045  https://cdn.myanimelist.net/r/50x70/images/man…
20046  https://cdn.myanimelist.net/r/50x70/images/man…
20047  https://cdn.myanimelist.net/r/50x70/images/man…
20048  https://cdn.myanimelist.net/r/50x70/images/man…
20049  https://cdn.myanimelist.net/r/50x70/images/man…

[20050 rows x 7 columns]
```

[238]:
```
merged_data[merged_data['Title'].str.contains('Yoake ni Furu,')]
```

[238]:
```
                Title                                              Link  \
1      Yoake ni Furu,  https://myanimelist.net/manga/57723/Yoake_ni_Furu
19751  Yoake ni Furu,  https://myanimelist.net/manga/57723/Yoake_ni_Furu

       Original Title English Title Genres  Score_y  \
1      Yoake ni Furu,           NaN    NaN     7.49
19751  Yoake ni Furu,           NaN    NaN     7.49
```

	Image Link
1	https://cdn.myanimelist.net/r/50x70/images/man…
19751	https://cdn.myanimelist.net/r/50x70/images/man…

manga-cleaning-EDA

December 10, 2024

[1]:
```
import pandas as pd
import numpy as np
```

[2]:
```
pd.reset_option('display.max_rows')
```

[3]:
```
df = pd.read_csv('final_merged_stats.csv')
df['links'][0]
```

[3]:
```
'https://myanimelist.net/manga/77079/Metto-kun_wa_Ikemen_desu'
```

[4]:
```
df2 = pd.read_csv('title-genre-synopsis-link.csv')
df2
```

[4]:
```
                                                   title  \
0                                                Horimiya
1                                                 Nisekoi
2                                            Ao Haru Ride
3                                                Bakuman.
4                                      5-toubun no Hanayome
…                                                       …
17195  Outaishi-sama, Watashi Kondo koso Anata ni Kor…
17196  Sayonara Elevator: Kowakute Setsunai Gakkou no…
17197                           Shiina-san no Oshi Jijou
17198                          Soshite Megami wa Hohoemu
17199                        Yomei Ichinen, Otoko wo Kau

                                genre  \
0                             Romance
1                      Comedy,Romance
2                             Romance
3                Comedy,Drama,Romance
4        Award Winning,Comedy,Romance
…                                  …
17195                Fantasy,Romance
17196    Drama,Romance,Supernatural
17197                 Comedy,Romance
17198                Fantasy,Romance
17199                         Romance
```

```
                                                  synopsis  \
0      Although admired at school for her amiability …
1      When Raku Ichijou was young, he made a heartfe…
2      While most girls desire popularity among boys,…
3      Despite being a talented artist, middle school…
4      Considered a genius, high schooler Fuutarou Ue…
…                                                    …
17195  The poverty-stricken noblewoman Liesel was mur…
17196  1-3. Sayonara Elevator\r\n4. Kimi ga Kureta Ki…
17197                                  (No synopsis yet.)
17198                                  (No synopsis yet.)
17199                                  (No synopsis yet.)

                                                   links
0            https://myanimelist.net/manga/42451/Horimiya
1             https://myanimelist.net/manga/31499/Nisekoi
2       https://myanimelist.net/manga/24294/Ao_Haru_Ride
3             https://myanimelist.net/manga/9711/Bakuman
4      https://myanimelist.net/manga/103851/5-toubun_…
…                                                    …
17195  https://myanimelist.net/manga/162121/Outaishi-…
17196  https://myanimelist.net/manga/140291/Sayonara_…
17197  https://myanimelist.net/manga/165492/Shiina-sa…
17198  https://myanimelist.net/manga/151856/Soshite_M…
17199  https://myanimelist.net/manga/157563/Yomei_Ich…

[17200 rows x 4 columns]
```

1 Merge Datasets

[6]:
```
# merge the two data frames on the links column
manga_info = df2.merge(df, how='inner', on='links')
# reorgamize the columns
manga_info = manga_info[['title', 'score', 'rank', 'popularity', 'type',␣
 ↪'genre', 'theme', 'favorites', 'vol', 'chap', 'status', 'published',␣
 ↪'synopsis', 'links']]
manga_info
```

[6]:
```
                                        title  \
0                                    Horimiya
1                                     Nisekoi
2                                Ao Haru Ride
3                                    Bakuman.
4                          5-toubun no Hanayome
…                                          …
```

```
17874  V-idol ni Motesugite Kurutta Boku no Nichijou
17875             Yorugata Hokeni no Adashino-sensei
17876                                Yukine Konohana
17877    Atsuki Shinou ni Chirasarete: Oushiza no Ai
17878                 Migawari Hanayome no Mitsugetsu

                         score    rank  popularity         type  \
0                         8.43     191          16        Manga
1                         7.72    1621          37        Manga
2                         8.14     511          41        Manga
3                         8.38     231          48        Manga
4                         7.93     894          56        Manga
…                          …     …           …           …
17874  N/A1 (scored by - users)  385422       69859        Manga
17875  N/A1 (scored by - users)  325732       68333        Manga
17876  N/A1 (scored by - users)  320172       69321     One-shot
17877  N/A1 (scored by - users)  421532       69184  Light Novel
17878  N/A1 (scored by - users)  420622       69479  Light Novel

                             genre          theme favorites      vol  \
0                          Romance         School    25,073     17.0
1                 Comedy,Romance   Harem,School    13,347     25.0
2                          Romance         School     8,929     13.0
3          Comedy,Drama,Romance  Otaku Culture    11,065     20.0
4      Award Winning,Comedy,Romance   Harem,School    10,740     14.0
…                              …            …       …       …
17874              Comedy,Romance        Unknown         0  Unknown
17875        Romance,Supernatural         School         0        1
17876                      Romance        Unknown         0  Unknown
17877                      Romance        Unknown         0        1
17878            Fantasy,Romance        Unknown         0        1

          chap      status                        published  \
0        139.0    Finished  Oct  18, 2011 to Mar  18, 2021
1        229.0    Finished    Nov  7, 2011 to Aug  8, 2016
2         53.0    Finished  Jan  13, 2011 to Feb  13, 2015
3        176.0    Finished  Aug  11, 2008 to Apr  23, 2012
4        122.0    Finished   Aug  9, 2017 to Feb  19, 2020
…        …         …                              …
17874  Unknown  Publishing               May 22, 2020 to ?
17875        4    Finished    May 24, 2019 to Aug 24, 2019
17876        1    Finished                    Dec 23, 2021
17877        8    Finished                    Feb 12, 2016
17878  Unknown    Finished                    Jan 25, 2016

                                              synopsis  \
0      Although admired at school for her amiability …
```

```
1      When Raku Ichijou was young, he made a heartfe…
2      While most girls desire popularity among boys,…
3      Despite being a talented artist, middle school…
4      Considered a genius, high schooler Fuutarou Ue…
…                                                    …
17874                                 (No synopsis yet.)
17875                                 (No synopsis yet.)
17876                                 (No synopsis yet.)
17877                                 (No synopsis yet.)
17878                                 (No synopsis yet.)

                                                   links
0           https://myanimelist.net/manga/42451/Horimiya
1            https://myanimelist.net/manga/31499/Nisekoi
2       https://myanimelist.net/manga/24294/Ao_Haru_Ride
3             https://myanimelist.net/manga/9711/Bakuman
4      https://myanimelist.net/manga/103851/5-toubun_…
…                                                    …
17874  https://myanimelist.net/manga/128827/V-idol_ni…
17875  https://myanimelist.net/manga/155523/Yorugata_…
17876  https://myanimelist.net/manga/152970/Yukine_Ko…
17877  https://myanimelist.net/manga/96648/Atsuki_Shi…
17878  https://myanimelist.net/manga/96360/Migawari_H…

[17879 rows x 14 columns]
```

[7]:
```
# see what the columns dtypes are
manga_info.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17879 entries, 0 to 17878
Data columns (total 14 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   title       17879 non-null  object
 1   score       17879 non-null  object
 2   rank        17879 non-null  object
 3   popularity  17879 non-null  int64
 4   type        17879 non-null  object
 5   genre       17879 non-null  object
 6   theme       17615 non-null  object
 7   favorites   17879 non-null  object
 8   vol         17504 non-null  object
 9   chap        17586 non-null  object
 10  status      17879 non-null  object
 11  published   17866 non-null  object
 12  synopsis    17862 non-null  object
 13  links       17879 non-null  object
```

```
dtypes: int64(1), object(13)
memory usage: 1.9+ MB
```

Cleaning procedures - we will need to change some of the columns dtypes - relace missing values with appropriate values

1. theme, vol, chap, published, and synipsis all have missing values that need to be addressed
2. score, rank, favorites, vol, and chap should all be int64 dtypes

[9]:
```
# There are a few columns to look through so I will make a loop to look at them all
for i in manga_info.columns:
    column_values = manga_info[i].isna().value_counts()
    print(column_values)
```

```
title
False    17879
Name: count, dtype: int64
score
False    17879
Name: count, dtype: int64
rank
False    17879
Name: count, dtype: int64
popularity
False    17879
Name: count, dtype: int64
type
False    17879
Name: count, dtype: int64
genre
False    17879
Name: count, dtype: int64
theme
False    17615
True       264
Name: count, dtype: int64
favorites
False    17879
Name: count, dtype: int64
vol
False    17504
True       375
Name: count, dtype: int64
chap
False    17586
True       293
Name: count, dtype: int64
status
```

```
False    17879
Name: count, dtype: int64
published
False    17866
True        13
Name: count, dtype: int64
synopsis
False    17862
True        17
Name: count, dtype: int64
links
False    17879
Name: count, dtype: int64
```

Theme - There are 264 null values in theme
Vol - There are 375 null values in vol
chap - There are 293 null values in chap
published - There are 13 null values in published
synopsis - There are 17 null values in synopsis

With how few null values are in pubished and synopsis we could get rid of those rows.
We can fill the vol and chap column with the median value as it wont be affected by outliers as much
Theme can be filled with Unknown

[11]:
```
# drop synopsis and published null rows
manga_info.dropna(subset = ['synopsis', 'published'], inplace=True)
```

[12]:
```
# check if null values have been dropped
print('Number of missing values in synopsis-', manga_info['synopsis'].isna().
  ↪sum())
print('Number of missing values in published-', manga_info['published'].isna().
  ↪sum())
```

```
Number of missing values in synopsis- 0
Number of missing values in published- 0
```

[13]:
```
manga_info[manga_info['synopsis'].str.contains('none')]['synopsis']
```

[13]:
```
4        Considered a genius, high schooler Fuutarou Ue…
12       There is nowhere that Seon Jin can find solace…
16       Natsuo Fujii is in love with his teacher, Hina…
138      The members of the Umino household are all rat…
353      Parallel to our dimension is a world filled wi…
                              …
16792    Emilia is the beloved daughter of a renowned E…
17396    June, a political reporter at a newspaper gian…
17568    Mariah doesn't trust men. It doesn't help that…
17628    Ami Sato works for a confectionery store, but …
```

```
17755    Mina is stunned when she sees a multimillionai…
Name: synopsis, Length: 147, dtype: object
```

There are duplicate manga that have different formats for how they are scored. One is in float format and one is in int format
One of these will get dropped and fomated to fit with the other values.

[15]:
```
manga_info[3464:3467]
```

[15]:
```
                                 title                          score  \
3474                       BxB Brothers                           7.38
3475  Shinomiya-kun no Sekai ga Owattemo  6.991 (scored by 1,135 users)
3476  Shinomiya-kun no Sekai ga Owattemo                           7.0

       rank  popularity   type                 genre   theme favorites   vol  \
3474   3920        7364  Manga  Comedy,Drama,Romance  Shoujo         3  10.0
3475  89532        7418  Manga               Romance  School         6     1
3476   8702        7373  Manga               Romance  School         6   1.0

      chap     status                     published  \
3474  70.0   Finished                  2001 to  2004
3475     5   Finished    Oct 11, 2014 to Dec 13, 2014
3476   5.0   Finished  Oct  11, 2014 to Dec  13, 2014

                                              synopsis  \
3474  Iizuka Sono just transferred to a new school w…
3475  Shinomiya-kun's aloofness attracted Yukino's a…
3476  Shinomiya-kun's aloofness attracted Yukino's a…

                                                 links
3474    https://myanimelist.net/manga/1077/BxB_Brothers
3475  https://myanimelist.net/manga/83153/Shinomiya-…
3476  https://myanimelist.net/manga/83153/Shinomiya-…
```

[16]:
```
manga_info.drop_duplicates(subset=['title', 'synopsis'], inplace=True)
manga_info[3464:3467]
```

[16]:
```
                                    title                          score  \
4874                     Rosario to Vampire  7.311 (scored by 469 users)
4875  The Villainess Becomes the Leading Lady  6.611 (scored by 464 users)
4876                Waltz wa Shiroi Dress de  7.191 (scored by 704 users)

       rank  popularity         type  \
4874  46592       11360  Light Novel
4875  145152       9448       Manhwa
4876  60732       11189        Manga

                                        genre               theme  \
```

```
4874  Action,Comedy,Fantasy,Romance,Supernatural       Harem, School
4875                      Drama,Fantasy,Romance  Isekai, Villainess
4876                             Drama,Romance          Historical

     favorites      vol chap    status                     published  \
4874        39        1    4  Finished                   Aug 4, 2008
4875         3  Unknown  125  Finished  Jun 10, 2021 to Jun 15, 2023
4876         6        4   19  Finished                          1990

                                              synopsis  \
4874                                (No synopsis yet.)
4875  Reborn into a fantasy novel as a wicked charac…
4876  At the beginning of the XX century, the relati…

                                                  links
4874  https://myanimelist.net/manga/90337/Rosario_to…
4875  https://myanimelist.net/manga/149108/The_Villa…
4876  https://myanimelist.net/manga/830/Waltz_wa_Shi…
```

[17]:
```
# reset index
manga_info.reset_index(drop = True)
```

[17]:
```
                                          title  \
0                                       Horimiya
1                                        Nisekoi
2                                  Ao Haru Ride
3                                      Bakuman.
4                             5-toubun no Hanayome
…                                              …
16275  V-idol ni Motesugite Kurutta Boku no Nichijou
16276            Yorugata Hokeni no Adashino-sensei
16277                                Yukine Konohana
16278   Atsuki Shinou ni Chirasarete: Oushiza no Ai
16279               Migawari Hanayome no Mitsugetsu

                        score    rank  popularity         type  \
0                        8.43     191          16        Manga
1                        7.72    1621          37        Manga
2                        8.14     511          41        Manga
3                        8.38     231          48        Manga
4                        7.93     894          56        Manga
…                           …       …           …           …
16275  N/A1 (scored by - users)  385422       69859        Manga
16276  N/A1 (scored by - users)  325732       68333        Manga
16277  N/A1 (scored by - users)  320172       69321     One-shot
16278  N/A1 (scored by - users)  421532       69184  Light Novel
16279  N/A1 (scored by - users)  420622       69479  Light Novel
```

```
                              genre          theme favorites      vol  \
0                           Romance         School    25,073     17.0
1                    Comedy,Romance   Harem,School    13,347     25.0
2                           Romance         School     8,929     13.0
3              Comedy,Drama,Romance  Otaku Culture    11,065     20.0
4      Award Winning,Comedy,Romance   Harem,School    10,740     14.0
…                               …            …       …       …
16275                Comedy,Romance        Unknown         0  Unknown
16276          Romance,Supernatural         School         0        1
16277                       Romance        Unknown         0  Unknown
16278                       Romance        Unknown         0        1
16279                Fantasy,Romance        Unknown         0        1

          chap      status                        published  \
0        139.0    Finished  Oct  18, 2011 to Mar  18, 2021
1        229.0    Finished    Nov  7, 2011 to Aug  8, 2016
2         53.0    Finished  Jan  13, 2011 to Feb  13, 2015
3        176.0    Finished  Aug  11, 2008 to Apr  23, 2012
4        122.0    Finished   Aug  9, 2017 to Feb  19, 2020
…          …         …                             …
16275  Unknown  Publishing               May 22, 2020 to ?
16276        4    Finished    May 24, 2019 to Aug 24, 2019
16277        1    Finished                    Dec 23, 2021
16278        8    Finished                    Feb 12, 2016
16279  Unknown    Finished                    Jan 25, 2016

                                              synopsis  \
0      Although admired at school for her amiability …
1      When Raku Ichijou was young, he made a heartfe…
2      While most girls desire popularity among boys,…
3      Despite being a talented artist, middle school…
4      Considered a genius, high schooler Fuutarou Ue…
…                                                    …
16275                                (No synopsis yet.)
16276                                (No synopsis yet.)
16277                                (No synopsis yet.)
16278                                (No synopsis yet.)
16279                                (No synopsis yet.)

                                                  links
0           https://myanimelist.net/manga/42451/Horimiya
1            https://myanimelist.net/manga/31499/Nisekoi
2       https://myanimelist.net/manga/24294/Ao_Haru_Ride
3             https://myanimelist.net/manga/9711/Bakuman
4      https://myanimelist.net/manga/103851/5-toubun_…
…                                                      …
```

```
16275  https://myanimelist.net/manga/128827/V-idol_ni…
16276  https://myanimelist.net/manga/155523/Yorugata_…
16277  https://myanimelist.net/manga/152970/Yukine_Ko…
16278  https://myanimelist.net/manga/96648/Atsuki_Shi…
16279  https://myanimelist.net/manga/96360/Migawari_H…

[16280 rows x 14 columns]
```

1.0.1 Fill missing values

[19]:
```
# replace non numeric values with np.nan
manga_info['chap'] = pd.to_numeric(manga_info['chap'], errors='coerce')
# find the median value for the volume column
chap_median = manga_info['chap'].median()
# fill nan values with the median value
manga_info['chap'].fillna(value = chap_median)
```

[19]:
```
0        139.0
1        229.0
2         53.0
3        176.0
4        122.0
         …
17874      9.0
17875      4.0
17876      1.0
17877      8.0
17878      9.0
Name: chap, Length: 16280, dtype: float64
```

[20]:
```
# replace non numeric values with np.nan
manga_info['vol'] = pd.to_numeric(manga_info['vol'], errors='coerce')
# find the median value for the volume column
vol_median = manga_info['vol'].median()
# fill nan values with the median value
manga_info['vol'].fillna(value = vol_median)
```

[20]:
```
0        17.0
1        25.0
2        13.0
3        20.0
4        14.0
         …
17874     1.0
17875     1.0
17876     1.0
17877     1.0
```

```
17878     1.0
Name: vol, Length: 16280, dtype: float64
```

[21]:
```
# check how many theme values are null
manga_info['theme'].isna().sum()
```

[21]:
```
118
```

[22]:
```
# fill nan values with Unknown
manga_info['theme'].fillna("Unknown", inplace = True)
# check if null values still exist
manga_info['theme'].isna().sum()
```

[22]:
```
0
```

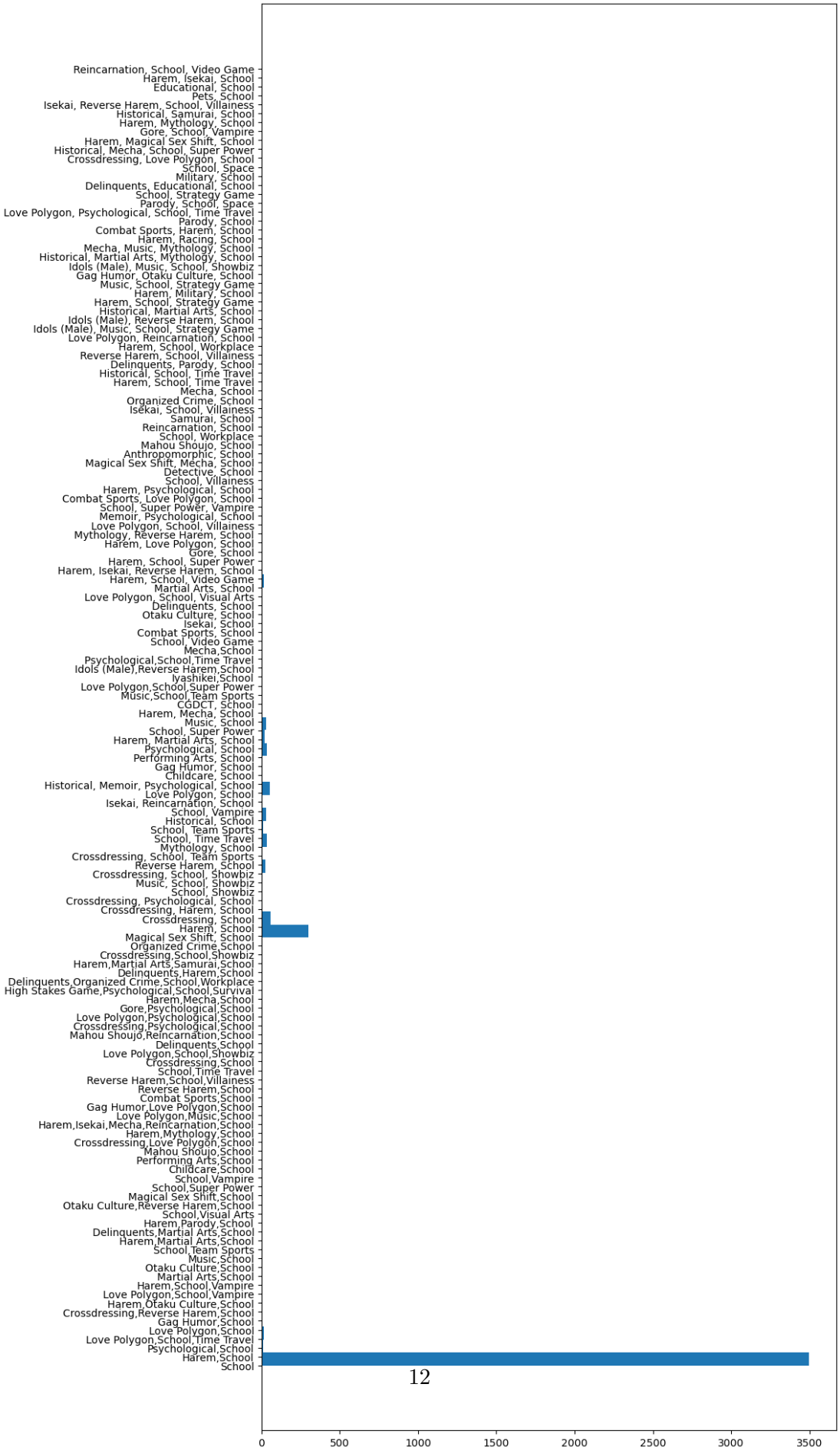
[23]:
```
manga_info.to_csv('all_manga_info_cleaned.csv')
```

[25]:
```
import matplotlib.pyplot as plt
```

[26]:
```
rom_school_theme = manga_info[(manga_info['theme'].str.contains('School')) &␣
 ↪(manga_info['genre'].str.contains('Romance'))]
rom_school_theme.value_counts().sum()
```

[26]:
```
2930
```

[27]:
```
plt.figure(figsize=(10, 25))  # This will make the plot 10 inches wide and 15␣
 ↪inches tall
plt.hist(rom_school_theme['theme'], orientation='horizontal', bins= 100)
plt.show()
```

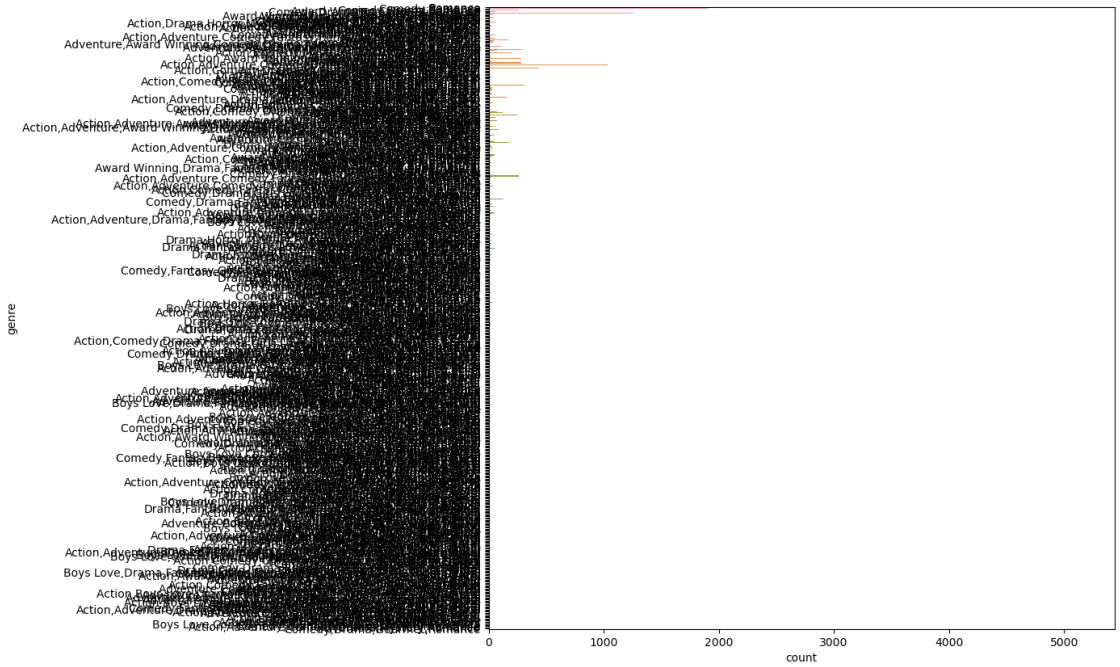

Reincarnation, School, Video Game
Harem, Isekai, School
Educational, School
Pets, School
Isekai, Reverse Harem, School, Villainess
Historical, Samurai, School
Harem, Mythology, School
Gore, School, Vampire
Harem, Magical Sex Shift, School
Historical, Mecha, School, Super Power
Crossdressing, Love Polygon, School
School, Space
Military, School
Delinquents, Educational, School
School, Strategy Game
Parody, School, Space
Love Polygon, Psychological, School, Time Travel
Parody, School
Combat Sports, Harem, School
Harem, Racing, School
Mecha, Music, Mythology, School
Historical, Martial Arts, Mythology, School
Idols (Male), Music, School, Showbiz
Gag Humor, Otaku Culture, School
Music, School, Strategy Game
Harem, Military, School
Harem, School, Strategy Game
Historical, Martial Arts, School
Idols (Male), Reverse Harem, School
Idols (Male), Music, School, Strategy Game
Love Polygon, Reincarnation, School
Harem, School, Workplace
Reverse Harem, School, Villainess
Delinquents, Parody, School
Historical, School, Time Travel
Harem, School, Time Travel
Mecha, School
Organized Crime, School
Isekai, School, Villainess
Samurai, School
Reincarnation, School
School, Workplace
Mahou Shoujo, School
Anthropomorphic, School
Magical Sex Shift, Mecha, School
Detective, School
School, Villainess
Harem, Psychological, School
Combat Sports, Love Polygon, School
School, Super Power, Vampire
Memoir, Psychological, School
Love Polygon, School, Villainess
Mythology, Reverse Harem, School
Harem, Love Polygon, School
Gore, School
Harem, School, Super Power
Harem, Isekai, Reverse Harem, School
Harem, School, Video Game
Martial Arts, School
Love Polygon, School, Visual Arts
Delinquents, School
Otaku Culture, School
Isekai, School
Combat Sports, School
School, Video Game
Mecha,School
Psychological,School,Time Travel
Idols (Male),Reverse Harem,School
Iyashikei,School
Love Polygon,School,Super Power
Music,School,Team Sports
CGDCT, School
Harem, Mecha, School
Music, School
School, Super Power
Harem, Martial Arts, School
Psychological, School
Performing Arts, School
Gag Humor, School
Childcare, School
Historical, Memoir, Psychological, School
Love Polygon, School
Isekai, Reincarnation, School
School, Vampire
Historical, School
School, Team Sports
School, Time Travel
Mythology, School
Crossdressing, School, Team Sports
Reverse Harem, School
Crossdressing, School, Showbiz
Music, School, Showbiz
School, Showbiz
Crossdressing, Psychological, School
Crossdressing, Harem, School
Crossdressing, School
Harem, School
Magical Sex Shift, School
Organized Crime,School
Crossdressing,School,Showbiz
Harem,Martial Arts,Samurai,School
Delinquents,Harem,School
Delinquents,Organized Crime,School,Workplace
High Stakes Game,Psychological,School,Survival
Harem,Mecha,School
Gore,Psychological,School
Love Polygon,Psychological,School
Crossdressing,Psychological,School
Mahou Shoujo,Reincarnation,School
Delinquents,School
Love Polygon,School,Showbiz
Crossdressing,School
School,Time Travel
Reverse Harem,School,Villainess
Reverse Harem,School
Combat Sports,School
Gag Humor,Love Polygon,School
Love Polygon,Music,School
Harem,Isekai,Mecha,Reincarnation,School
Harem,Mythology,School
Crossdressing,Love Polygon,School
Mahou Shoujo,School
Performing Arts,School
Childcare,School
School,Vampire
School,Super Power
Magical Sex Shift,School
Otaku Culture,Reverse Harem,School
School,Visual Arts
Harem,Parody,School
Delinquents,Martial Arts,School
Harem,Martial Arts,School
School,Team Sports
Music,School
Otaku Culture,School
Martial Arts,School
Harem,School,Vampire
Love Polygon,School,Vampire
Harem,Otaku Culture,School
Crossdressing,Reverse Harem,School
Gag Humor,School
Love Polygon,School
Love Polygon,School,Time Travel
Psychological,School
Harem,School
School
0
500
1000
1500
2000
2500
3000
3500

Most of the romantic school manga fall within “Harem,School” and “School”

[29]:
```
import seaborn as sns
```

[30]:
```
plt.figure(figsize=(10,10))
sns.countplot(data=manga_info, y='genre')
```

[30]:
```
<Axes: xlabel='count', ylabel='genre'>
```



[31]:
```
# pd.set_option('display.max_rows', None)
pd.reset_option('display.max_rows')
genres_list = list(manga_info['genre'])
```

2 Genre Mapping

[33]:
```
genres = genres_list

genre_mapping = {
    'Ecchi': 'Ecchi/Erotica',
    'Erotica': 'Ecchi/Erotica',
    'Supernatural': 'Fantasy/Supernatural',
    'Fantasy': 'Fantasy/Supernatural',
```

```
    'Horror': 'Horror/Thriller',
    'Mystery': 'Horror/Thriller',
    'Suspense': 'Horror/Thriller',
    'Slice of Life': 'Slice of Life',
    'Romance': 'Romance',
    'Comedy': 'Comedy',
    'Sci-Fi': 'Sci-Fi',
    'Sports': 'Sports',
    'Adventure': 'Action/Adventure',
    'Action': 'Action/Adventure',
    'Boys Love': 'Romance',
    'Girls Love': 'Romance',
}

# Function to map genres to broader categories
def map_genres(genre_str):
    for key, category in genre_mapping.items():
        if key in genre_str:
            return category
    return 'Other'  # Default for genres that don't fit into a broad category

# Apply the function to the list of genres
aggregated_genres = [map_genres(genre) for genre in genres]

manga_info['agg_genres'] = aggregated_genres
manga_info
```

[33]:

	title	\
0	Horimiya	
1	Nisekoi	
2	Ao Haru Ride	
3	Bakuman.	
4	5-toubun no Hanayome	
…	…	
17874	V-idol ni Motesugite Kurutta Boku no Nichijou	
17875	Yorugata Hokeni no Adashino-sensei	
17876	Yukine Konohana	
17877	Atsuki Shinou ni Chirasarete: Oushiza no Ai	
17878	Migawari Hanayome no Mitsugetsu	

	score	rank	popularity	type	\
0	8.43	191	16	Manga	
1	7.72	1621	37	Manga	
2	8.14	511	41	Manga	
3	8.38	231	48	Manga	
4	7.93	894	56	Manga	
…	…	…	…	…	

```
17874  N/A1 (scored by - users)  385422       69859        Manga
17875  N/A1 (scored by - users)  325732       68333        Manga
17876  N/A1 (scored by - users)  320172       69321     One-shot
17877  N/A1 (scored by - users)  421532       69184  Light Novel
17878  N/A1 (scored by - users)  420622       69479  Light Novel

                            genre          theme favorites   vol   chap  \
0                         Romance         School    25,073  17.0  139.0
1                 Comedy,Romance   Harem,School    13,347  25.0  229.0
2                         Romance         School     8,929  13.0   53.0
3            Comedy,Drama,Romance  Otaku Culture    11,065  20.0  176.0
4      Award Winning,Comedy,Romance   Harem,School    10,740  14.0  122.0
…                              …              …       …    …    …
17874             Comedy,Romance        Unknown         0   NaN    NaN
17875       Romance,Supernatural         School         0   1.0    4.0
17876                     Romance        Unknown         0   NaN    1.0
17877                     Romance        Unknown         0   1.0    8.0
17878           Fantasy,Romance        Unknown         0   1.0    NaN

          status                        published  \
0       Finished  Oct  18, 2011 to Mar  18, 2021
1       Finished    Nov  7, 2011 to Aug  8, 2016
2       Finished  Jan  13, 2011 to Feb  13, 2015
3       Finished  Aug  11, 2008 to Apr  23, 2012
4       Finished   Aug  9, 2017 to Feb  19, 2020
…          …                             …
17874  Publishing               May 22, 2020 to ?
17875   Finished    May 24, 2019 to Aug 24, 2019
17876   Finished                   Dec 23, 2021
17877   Finished                   Feb 12, 2016
17878   Finished                   Jan 25, 2016

                                              synopsis  \
0      Although admired at school for her amiability …
1      When Raku Ichijou was young, he made a heartfe…
2      While most girls desire popularity among boys,…
3      Despite being a talented artist, middle school…
4      Considered a genius, high schooler Fuutarou Ue…
…                                                    …
17874                                (No synopsis yet.)
17875                                (No synopsis yet.)
17876                                (No synopsis yet.)
17877                                (No synopsis yet.)
17878                                (No synopsis yet.)

                                                 links            agg_genres
0           https://myanimelist.net/manga/42451/Horimiya               Romance
```

```
1            https://myanimelist.net/manga/31499/Nisekoi                Romance
2       https://myanimelist.net/manga/24294/Ao_Haru_Ride                Romance
3             https://myanimelist.net/manga/9711/Bakuman                Romance
4      https://myanimelist.net/manga/103851/5-toubun_…                 Romance
…                                                    …                    …
17874  https://myanimelist.net/manga/128827/V-idol_ni…                 Romance
17875  https://myanimelist.net/manga/155523/Yorugata_…  Fantasy/Supernatural
17876  https://myanimelist.net/manga/152970/Yukine_Ko…                 Romance
17877  https://myanimelist.net/manga/96648/Atsuki_Shi…                 Romance
17878  https://myanimelist.net/manga/96360/Migawari_H…  Fantasy/Supernatural

[16280 rows x 15 columns]
```

[93]:
```python
#pd.set_option('display.max_rows', None)
# pd.reset_option('display.max_rows')
theme_list = list(manga_info['theme'])

themes = theme_list

theme_mapping = {
    "Horror": [
        'Gore', 'Psychological', 'Survival', 'Vampire', 'Mythology', 'High␣
 ↪Stakes Game',
        'Gore, Psychological', 'Gore, Survival', 'Gore, School', 'Gore,␣
 ↪Organized Crime',
        'Gore, Psychological, Visual Arts', 'Gore, School, Vampire'
    ],
    "Sci-Fi": [
        'Mecha', 'Space', 'Super Power', 'Time Travel', 'Strategy Game',␣
 ↪'Anthropomorphic', 'Isekai',
        'Mecha, Military', 'Mecha, Space', 'Mecha, Psychological', 'Mecha,␣
 ↪Music, Mythology, School',
        'Space, Vampire', 'Mecha, Military, Space', 'Isekai, School', 'Isekai,␣
 ↪Time Travel',
        'Isekai, Villainess', 'Isekai, Medical', 'Mecha, Space, Super Power'
    ],
    "Romance": [
        'Love Polygon', 'Reverse Harem', 'Villainess', 'Crossdressing',␣
 ↪'Magical Sex Shift', 'Harem',
        'Love Polygon, Mythology', 'Harem, School', 'Harem, Reincarnation',␣
 ↪'Love Polygon, Psychological',
        'Crossdressing, Love Polygon, School', 'Harem, Love Polygon, School',␣
 ↪'Love Status Quo'
    ],
    "Humor": [
```

```
        'Gag Humor', 'Parody', 'Otaku Culture', 'Gag Humor, Otaku Culture,␣
↪School', 'Gag Humor, School',
        'Parody, School', 'Gag Humor, Love Polygon, School'
    ],
    "Drama": [
        'Adult Cast', 'Workplace', 'Historical', 'Showbiz', 'Visual Arts',␣
↪'Performing Arts',
        'Adult Cast, Love Polygon', 'Adult Cast, Workplace', 'Adult Cast,␣
↪Reincarnation',
        'Historical, Psychological', 'Adult Cast, Showbiz, Vampire',␣
↪'Performing Arts, School',
        'Adult Cast, Otaku Culture, Workplace', 'Historical, Organized Crime'
    ],
    "Fantasy": [
        'Mahou Shoujo', 'Mythology', 'Reincarnation', 'Samurai', 'Idols␣
↪(Male)', 'Idols (Female)',
        'Mahou Shoujo, Reverse Harem', 'Reincarnation, Time Travel', 'Mahou␣
↪Shoujo, School',
        'Mythology, Reincarnation', 'Isekai, Magical Sex Shift'
    ],
    "Action/Adventure": [
        'Martial Arts', 'Delinquents', 'Combat Sports', 'Racing', 'Military',␣
↪'Educational',
        'Delinquents, School', 'Delinquents, Racing', 'Martial Arts, Mecha,␣
↪Space, Super Power',
        'Military, School', 'Combat Sports, Love Polygon', 'Delinquents,␣
↪Organized Crime, School, Workplace'
    ],
    "Slice of Life": [
        'School', 'Childcare', 'CGDCT', 'Iyashikei', 'Childcare, School',␣
↪'School, Team Sports',
        'Iyashikei, Psychological, Visual Arts', 'School, Visual Arts',␣
↪'School, Workplace'
    ],
    "Mystery/Thriller": [
        'Detective', 'Psychological, Time Travel', 'Detective, School',␣
↪'Detective, Harem',
        'Detective, Isekai, Reincarnation', 'Detective, Historical'
    ],
    "Music": [
        'Music', 'Music, School', 'Music, School, Showbiz', 'Music, Showbiz',␣
↪'Music, Psychological',
        'Music, Time Travel', 'Music, Reverse Harem, Showbiz'
    ],
    "Sports": [
```

```
        'Team Sports', 'School, Team Sports', 'Music, School, Team Sports',␣
 ↪'Combat Sports, School'
    ],
    "Medical": [
        'Medical', 'Medical, Time Travel', 'Anthropomorphic, Medical'
    ],
    "Historical Fiction": [
        'Historical, Samurai', 'Historical, Time Travel', 'Historical, School',␣
 ↪'Historical, Music',
        'Historical, Military, Music', 'Historical, Memoir, Psychological,␣
 ↪School', 'Historical, Reincarnation'
    ],
    "Miscellaneous": [
        'Unknown', 'Seinen', 'Shoujo', 'Shounen', 'Pets', 'Pets, Psychological'
    ]
}

# Function to map genres to broader categories
def map_themes(theme_str):
    for key, category in theme_mapping.items():
        if any(keyword in theme_str for keyword in category):
            return key
    return  'Uncategorized'

# Apply the function to the list of genres
manga_info['agg_themes'] = manga_info['theme'].apply(map_themes)
#manga_info['favorites'] = manga_info['favorites'].str.replace(',', '').
 ↪astype(int)
manga_info['score'] = pd.to_numeric(manga_info['score'], errors='coerce')
manga_info['score'].fillna(0, inplace=True)
manga_info
```

[93]:

	Unnamed: 0		title	score	\
0	0		Horimiya	8.43	
1	1		Nisekoi	7.72	
2	2		Ao Haru Ride	8.14	
3	3		Bakuman.	8.38	
4	4		5-toubun no Hanayome	7.93	
…	…		…	…	
16275	17874	V-idol ni Motesugite Kurutta Boku no Nichijou		0.00	
16276	17875	Yorugata Hokeni no Adashino-sensei		0.00	
16277	17876	Yukine Konohana		0.00	
16278	17877	Atsuki Shinou ni Chirasarete: Oushiza no Ai		0.00	
16279	17878	Migawari Hanayome no Mitsugetsu		0.00	

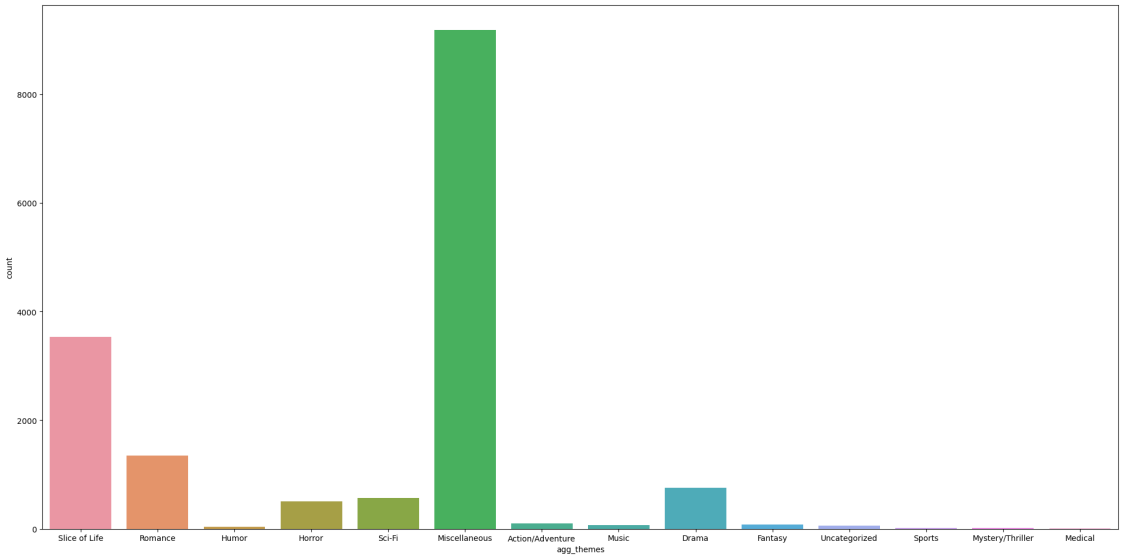
	rank	popularity	type	genre	\
0	191	16	Manga	Romance	

```
1         1621          37        Manga                Comedy,Romance
2          511          41        Manga                       Romance
3          231          48        Manga         Comedy,Drama,Romance
4          894          56        Manga  Award Winning,Comedy,Romance
…           …           …           …                             …
16275   385422       69859        Manga                Comedy,Romance
16276   325732       68333        Manga          Romance,Supernatural
16277   320172       69321     One-shot                       Romance
16278   421532       69184  Light Novel                       Romance
16279   420622       69479  Light Novel               Fantasy,Romance

                theme favorites   vol   chap      status  \
0              School    25,073  17.0  139.0    Finished
1       Harem,School    13,347  25.0  229.0    Finished
2              School     8,929  13.0   53.0    Finished
3      Otaku Culture    11,065  20.0  176.0    Finished
4       Harem,School    10,740  14.0  122.0    Finished
…                 …       …   …    …          …
16275        Unknown         0   NaN    NaN  Publishing
16276         School         0   1.0    4.0    Finished
16277        Unknown         0   NaN    1.0    Finished
16278        Unknown         0   1.0    8.0    Finished
16279        Unknown         0   1.0    NaN    Finished

                            published  \
0      Oct  18, 2011 to Mar  18, 2021
1        Nov  7, 2011 to Aug  8, 2016
2      Jan  13, 2011 to Feb  13, 2015
3      Aug  11, 2008 to Apr  23, 2012
4       Aug  9, 2017 to Feb  19, 2020
…                                  …
16275              May 22, 2020 to ?
16276    May 24, 2019 to Aug 24, 2019
16277                   Dec 23, 2021
16278                   Feb 12, 2016
16279                   Jan 25, 2016

                                                    synopsis  \
0      Although admired at school for her amiability …
1      When Raku Ichijou was young, he made a heartfe…
2      While most girls desire popularity among boys,…
3      Despite being a talented artist, middle school…
4      Considered a genius, high schooler Fuutarou Ue…
…                                                          …
16275                                 (No synopsis yet.)
16276                                 (No synopsis yet.)
16277                                 (No synopsis yet.)
```

```
16278                                (No synopsis yet.)
16279                                (No synopsis yet.)

                                                   links  \
0           https://myanimelist.net/manga/42451/Horimiya
1            https://myanimelist.net/manga/31499/Nisekoi
2       https://myanimelist.net/manga/24294/Ao_Haru_Ride
3             https://myanimelist.net/manga/9711/Bakuman
4      https://myanimelist.net/manga/103851/5-toubun_…
…                                                      …
16275  https://myanimelist.net/manga/128827/V-idol_ni…
16276  https://myanimelist.net/manga/155523/Yorugata_…
16277  https://myanimelist.net/manga/152970/Yukine_Ko…
16278  https://myanimelist.net/manga/96648/Atsuki_Shi…
16279  https://myanimelist.net/manga/96360/Migawari_H…

                  agg_genre     agg_themes            agg_genres
0                   Romance  Slice of Life               Romance
1                   Romance        Romance               Romance
2                   Romance  Slice of Life               Romance
3                   Romance          Humor               Romance
4                   Romance        Romance               Romance
…                     …              …                     …
16275               Romance  Miscellaneous               Romance
16276  Fantasy/Supernatural  Slice of Life  Fantasy/Supernatural
16277               Romance  Miscellaneous               Romance
16278               Romance  Miscellaneous               Romance
16279  Fantasy/Supernatural  Miscellaneous  Fantasy/Supernatural

[16280 rows x 18 columns]
```

[35]:
```
# manga_info = pd.read_csv('cleaned_book_info.csv')
```

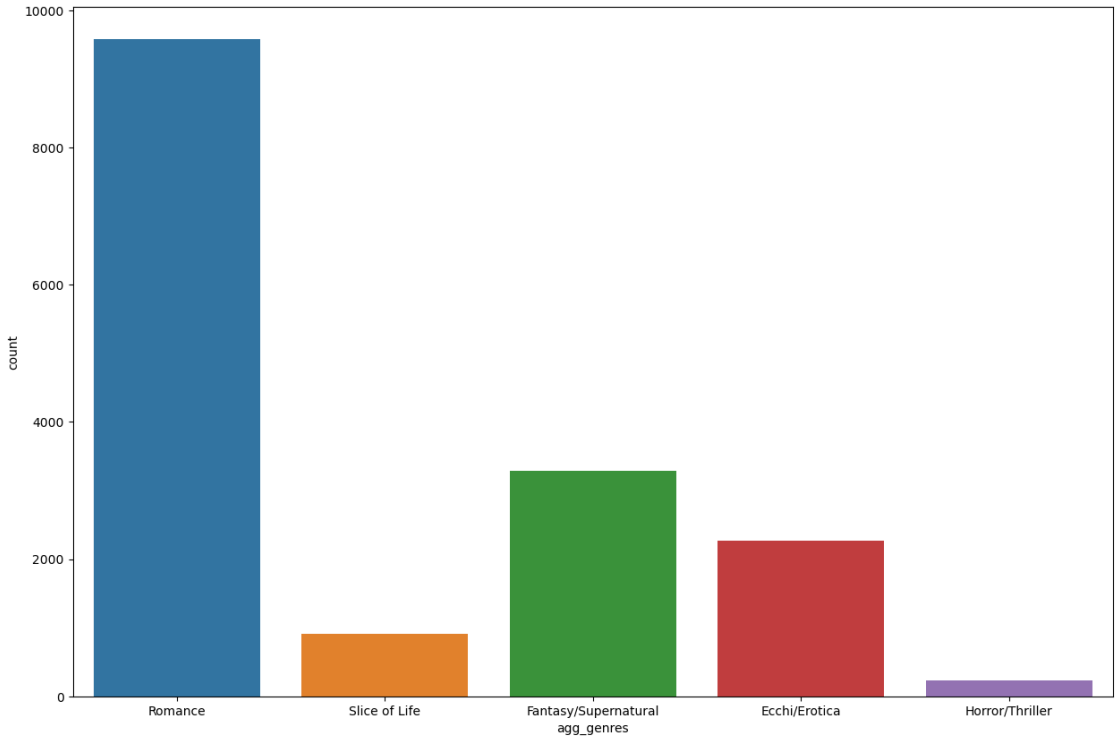
[114]:
```
plt.figure(figsize=(20,10))
sns.countplot(data=manga_info, x='agg_themes')
plt.tight_layout()
```



We can see in the graph above that a good amount of manga are uncategorized and might not be the most useful to use in our predictions. Using NLP methods to look at the synopsis and pull a theme from it might be possible but it looks like a lot of them uncategorized manga are also missing a synopsis.

[97]:
```
plt.figure(figsize=(15,10))
sns.countplot(data=manga_info, x='agg_genres')
```

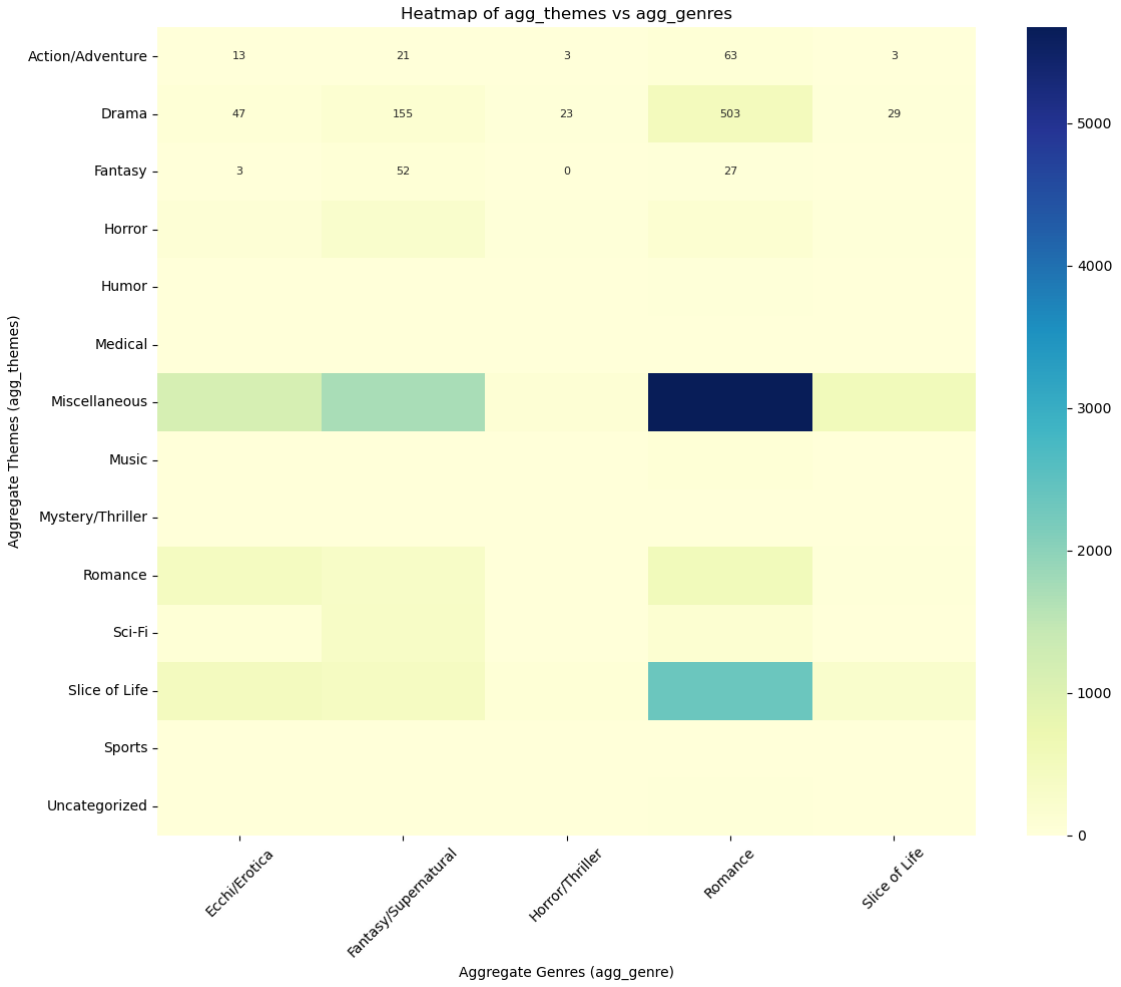
[97]:
```
<Axes: xlabel='agg_genres', ylabel='count'>
```



we have a full dataset of genres to work with and most of the books are romance related. Those could mix into other genes as well but they have a main component of romance.

3 Heatmap of the genres and themes

[110]:
```
# Create a contingency table comparing agg_themes and agg_genres
contingency_table = manga_info.pivot_table(index='agg_themes',␣
 ↪columns='agg_genre', aggfunc='size', fill_value=0)

# Generate the heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(contingency_table, annot=True, fmt="d", cmap="YlGnBu",␣
 ↪cbar=True,annot_kws={"size": 8})
plt.title('Heatmap of agg_themes vs agg_genres')
plt.xlabel('Aggregate Genres (agg_genre)')
plt.ylabel('Aggregate Themes (agg_themes)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



4 Highest Rated Genres

[124]:
```
non_zero = manga_info[manga_info['score'] > 0]

# Converting necessary columns to numeric for analysis
non_zero['score'] = pd.to_numeric(non_zero['score'], errors='coerce')
non_zero['rank'] = pd.to_numeric(non_zero['rank'], errors='coerce')
non_zero['popularity'] = pd.to_numeric(non_zero['popularity'], errors='coerce')

# Aggregating the data to analyze the most popular genres
genre_analysis = non_zero.groupby('agg_genre').agg(
    average_score=('score', 'mean'),
    median_rank=('rank', 'median'),
    total_entries=('agg_genre', 'count')
).sort_values(by='total_entries', ascending=False)
```

```
C:\Users\taylo\AppData\Local\Temp\ipykernel_37608\343521801.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  non_zero['score'] = pd.to_numeric(non_zero['score'], errors='coerce')
C:\Users\taylo\AppData\Local\Temp\ipykernel_37608\343521801.py:5:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  non_zero['rank'] = pd.to_numeric(non_zero['rank'], errors='coerce')
C:\Users\taylo\AppData\Local\Temp\ipykernel_37608\343521801.py:6:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  non_zero['popularity'] = pd.to_numeric(non_zero['popularity'],
errors='coerce')
```

[126]:
```
genre_analysis
```

[126]:

agg_genre	average_score	median_rank	total_entries
Romance	7.397177	3939.0	464
Fantasy/Supernatural	7.544857	2437.0	245
Ecchi/Erotica	7.195644	5119.5	202
Slice of Life	7.484930	3234.0	71
Horror/Thriller	7.292667	3205.0	15

[130]:
```
non_zero['score'].describe()
```

[130]:
```
count    997.000000
mean       7.397312
std        0.565327
min        5.310000
25%        7.050000
50%        7.410000
75%        7.770000
max        8.920000
Name: score, dtype: float64
```

4.0.1 What manga do users prefer and what is their average score.

[141]:
```
user_data = pd.read_csv('cleaned_user_data_v2.csv')
user_data.drop('Unnamed: 0', axis=1, inplace=True)
```

[172]:
```
# Aggregating the data to analyze the most popular genres
user_analysis = user_data.groupby('title').agg(
    average_score=('score', 'mean'),
    total_entries=('title', 'count')
).sort_values(by=['total_entries', 'average_score'], ascending=False)
```

[174]:
```
user_analysis.head(10)
```

[174]:

title	average_score	total_entries
Solo Leveling	8.317971	10372
Chainsaw Man	8.418749	9686
Berserk	9.302811	8573
Demon Slayer: Kimetsu no Yaiba	7.790927	7098
Attack on Titan	8.253152	6901
Jujutsu Kaisen	7.813131	6732
One Piece	9.079764	6607
Tokyo Ghoul	8.263667	6512
Horimiya	8.170419	5997
One-Punch Man	8.547644	5688

4.0.2 Does the vol or chap have an influence on the score

[155]:
```
# Converting `vol` and `chap` columns to numeric for analysis
manga_info['vol'] = pd.to_numeric(manga_info['vol'], errors='coerce')
manga_info['chap'] = pd.to_numeric(manga_info['chap'], errors='coerce')

# Exploring the relationship between chapters, volumes, and scores
engagement_analysis = non_zero[['score', 'vol', 'chap']].copy()

# Correlation analysis to determine relationships
correlations = engagement_analysis.corr()

# Displaying correlations for user
correlations
```

[155]:

	score	vol	chap
score	1.000000	0.401204	0.283945
vol	0.401204	1.000000	0.834223
chap	0.283945	0.834223	1.000000

These findings suggest that the number of volumes has a stronger influence on user ratings compared to the number of chapters. But this is still a somewhat weak correlation.

KNN

December 10, 2024

[8]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, MinMaxScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.neural_network import MLPClassifier
from sklearn.linear_model import LogisticRegression
```

[10]:
```
merged_data = pd.read_csv('user_book_merged.csv')
merged_data.drop('Unnamed: 0', axis=1, inplace=True)
```

[12]:
```
merged_data
```

[12]:

	user	user_score	\
0	RakaanG	7.0	
1	Marioobros	10.0	
2	AlainWakbar	6.0	
3	Amadyura	7.0	
4	Kurumim	7.0	
…	…	…	
240916	Kurakero	6.0	
240917	Kurakero	6.0	
240918	Snomnomnom	7.0	
240919	Diabla-Grace	6.0	
240920	GingerTenguin	7.0	

	title	avg_score	rank	\
0	Kitaku Tochuu de Yome to Musume ga Dekita n da…	6.9	9786	
1	Kitaku Tochuu de Yome to Musume ga Dekita n da…	6.9	9786	
2	Kitaku Tochuu de Yome to Musume ga Dekita n da…	6.9	9786	
3	Kitaku Tochuu de Yome to Musume ga Dekita n da…	6.9	9786	
4	Kitaku Tochuu de Yome to Musume ga Dekita n da…	6.9	9786	
…	…	…	…	
240916	Love & Tears	0.0	214192	
240917	Chocotan!	0.0	259202	
240918	Nibun no Ichi Boyfriend	0.0	482472	

```
240919  ACCA: 13-ku Kansatsu-ka Gaiden - Paula to Mich…        0.0  268052
240920                                  Ane no Tomodachi        7.4   36192

        popularity   type            genre              theme  favorites  \
0             4771  Manga  Fantasy,Romance            Unknown        6.0
1             4771  Manga  Fantasy,Romance            Unknown        6.0
2             4771  Manga  Fantasy,Romance            Unknown        6.0
3             4771  Manga  Fantasy,Romance            Unknown        6.0
4             4771  Manga  Fantasy,Romance            Unknown        6.0
…              …    …              …                  …        …
240916       31589  Manga    Drama,Romance            Unknown        0.0
240917       25393  Manga          Romance               Pets        2.0
240918       38878  Manga   Comedy,Romance  Magical Sex Shift        1.0
240919       28573  Manga    Drama,Romance            Unknown        1.0
240920        3058  Manga          Romance            Unknown       52.0

         vol  chap      status                        published  \
0        2.0  14.0    Finished  Jul  27, 2018 to Oct  15, 2019
1        2.0  14.0    Finished  Jul  27, 2018 to Oct  15, 2019
2        2.0  14.0    Finished  Jul  27, 2018 to Oct  15, 2019
3        2.0  14.0    Finished  Jul  27, 2018 to Oct  15, 2019
4        2.0  14.0    Finished  Jul  27, 2018 to Oct  15, 2019
…      …   …          …                             …
240916   1.0   NaN    Finished                      Not availa
240917  13.0  85.0    Finished      Oct 3, 2011 to Dec 1, 2017
240918   1.0   NaN   On Hiatus     Jan 8, 2016 to Mar 26, 2018
240919   1.0   6.0    Finished                   Dec 25, 2017
240920   NaN   NaN  Publishing               Jul 19, 2022 to ?

                                                synopsis  \
0       High school student Kazamachi Kunpei suddenly …
1       High school student Kazamachi Kunpei suddenly …
2       High school student Kazamachi Kunpei suddenly …
3       High school student Kazamachi Kunpei suddenly …
4       High school student Kazamachi Kunpei suddenly …
…                                                     …
240916           Collection of sentimental love stories.
240917  This story is about the dog named Chokotan who…
240918  Soujirou possesses a special predisposition th…
240919  The manga tells the story of two rival Jumoku …
240920                              (No synopsis yet.)

                                                  links  \
0       https://myanimelist.net/manga/114924/Kitaku_To…
1       https://myanimelist.net/manga/114924/Kitaku_To…
2       https://myanimelist.net/manga/114924/Kitaku_To…
3       https://myanimelist.net/manga/114924/Kitaku_To…
```

```
4       https://myanimelist.net/manga/114924/Kitaku_To…
…                                                     …
240916   https://myanimelist.net/manga/12420/Love___Tears
240917       https://myanimelist.net/manga/45237/Chocotan
240918  https://myanimelist.net/manga/101893/Nibun_no_…
240919  https://myanimelist.net/manga/110453/ACCA__13-…
240920  https://myanimelist.net/manga/157378/Ane_no_To…

                  agg_genre     agg_themes  genre_encoded  theme_encoded
0       Fantasy/Supernatural  Uncategorized              1              8
1       Fantasy/Supernatural  Uncategorized              1              8
2       Fantasy/Supernatural  Uncategorized              1              8
3       Fantasy/Supernatural  Uncategorized              1              8
4       Fantasy/Supernatural  Uncategorized              1              8
…                          …             …              …              …
240916               Romance  Uncategorized              3              8
240917               Romance  Uncategorized              3              8
240918               Romance        Romance              3              5
240919               Romance  Uncategorized              3              8
240920               Romance  Uncategorized              3              8

[240921 rows x 20 columns]
```

[14]:
```
# create a target variable were we can identify what is a good book and what is␣
 ↪not
merged_data['target'] = merged_data['user_score'].apply(lambda x: 0 if x < 6␣
 ↪else 1)
```

[16]:
```
# handel missing data
merged_data['chap'].fillna(0.0, inplace=True)
merged_data['vol'].fillna(0.0, inplace=True)
```

1 Data Splitting

[18]:
```
from sklearn.model_selection import train_test_split

# Splitting data
X = merged_data.drop(['target', 'genre', 'theme', 'user', 'links', 'synopsis',␣
 ↪'published', 'user_score', 'avg_score'], axis=1)
y = merged_data['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[20]:
```
categorical_features = ['agg_genre', 'agg_themes', 'type', 'title', 'status'] ␣
 ↪# Replace with actual categorical columns
```

```
numerical_features = ['chap', 'rank', 'popularity','vol', 'favorites']  #␣
 ↪Replace with actual numerical columns
```

2 Feature Encoding

[22]:
```
from sklearn.preprocessing import MinMaxScaler, OneHotEncoder
encoder = OneHotEncoder(sparse_output=False, handle_unknown='ignore')
X_train_encoded = encoder.fit_transform(X_train[categorical_features])
y_train_encoded = encoder.transform(X_test[categorical_features])
```

3 Data Normalization

[24]:
```
from sklearn.preprocessing import MinMaxScaler

# scale the data
scaler = MinMaxScaler()
X_train_scaled = scaler.fit_transform(merged_data[numerical_features])
```

4 Preprocessing

4.1 - One Hot Encoding categorical features

- We want the text to be recognized as numbers so the model can preform calculations on them ## - Min Max Scaling Numerical Features
- We want to numerical values to be scaled so their different proportions don't conflict with each other

[26]:
```
preprocessor = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_features),
        ('num', MinMaxScaler(), numerical_features)
    ])
```

5 KNN Pipeline

[28]:
```
pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('classifier', KNeighborsClassifier(n_neighbors=5))
])
```

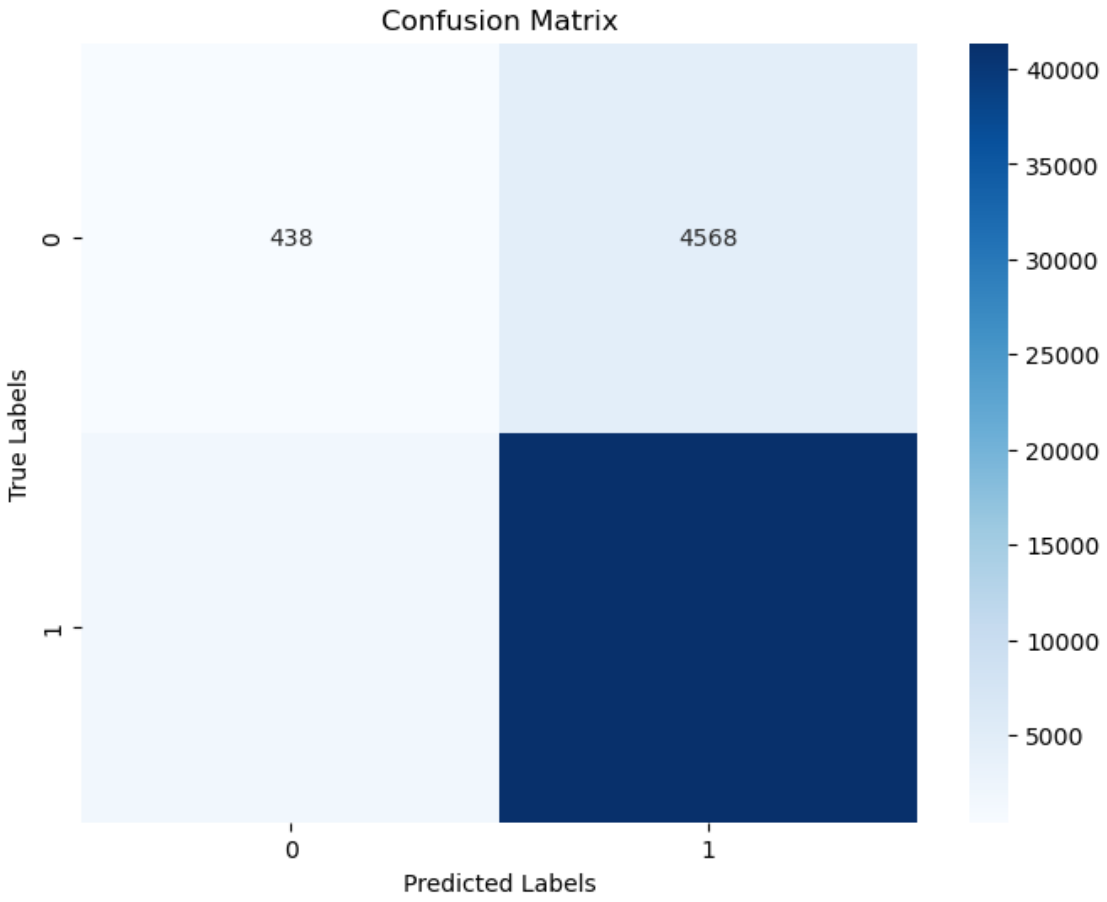
6 Model Prediction and Evaluation

[30]:
```
pipeline.fit(X_train, y_train)

# Test the Model
y_test_pred = pipeline.predict(X_test)
test_accuracy = accuracy_score(y_test, y_test_pred)
print(f"Test Accuracy: {test_accuracy:.2f}")
print("\nClassification Report on Test Data:")
print(classification_report(y_test, y_test_pred))
```

```
Test Accuracy: 0.88

Classification Report on Test Data:
              precision    recall  f1-score   support

           0       0.25      0.06      0.10      5006
           1       0.90      0.98      0.94     43179

    accuracy                           0.88     48185
   macro avg       0.57      0.52      0.52     48185
weighted avg       0.83      0.88      0.85     48185
```

7 Confusion matrix

[20]:
```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Compute confusion matrix
conf_matrix = confusion_matrix(y_test, y_test_pred)

# Plot confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',␣
 ↪xticklabels=pipeline.classes_, yticklabels=pipeline.classes_)
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.show()
```



8 MAE/RMSE Scores

[32]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np

# Assuming predictions are for continuous ratings
mae = mean_absolute_error(y_test, y_test_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_test_pred))

print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")
```

```
Mean Absolute Error (MAE): 0.12
Root Mean Squared Error (RMSE): 0.34
```

KNN gives us a very accurate model to work with.
Its not perfect which is good, we don't want it to be overfitting the data.
Lets look at other models and see how they compare.

9 Logistic Regression

[25]:
```
log_reg = LogisticRegression(max_iter=1000)
```

[26]:
```
lr_pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('classifier', log_reg)
])
```

[27]:
```
lr_pipeline.fit(X_train, y_train)
y_test_pred_lr = lr_pipeline.predict(X_test)
test_accuracy_lr = accuracy_score(y_test, y_test_pred_lr)
print(f"\nLogistic Regression Test Accuracy: {test_accuracy_lr:.2f}")
print("\nClassification Report (Logistic Regression):")
print(classification_report(y_test, y_test_pred_lr))
```

```
Logistic Regression Test Accuracy: 0.90

Classification Report (Logistic Regression):
              precision    recall  f1-score   support

           0       0.57      0.01      0.02      5006
           1       0.90      1.00      0.95     43179

    accuracy                           0.90     48185
   macro avg       0.73      0.51      0.48     48185
weighted avg       0.86      0.90      0.85     48185
```

We are really only looking for high scores on predicting good books.
We we can see that accuracy is in 90% as well as precision which looks at how many true positives did we got.

10 Neural Net

[34]:
```
neural_net = MLPClassifier(hidden_layer_sizes=(64, 32), max_iter=10,␣
↪random_state=42)
```

[36]:
```
nn_pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('classifier', neural_net)
])
```

[38]:
```
nn_pipeline.fit(X_train, y_train)
y_test_pred_nn = nn_pipeline.predict(X_test)
test_accuracy_nn = accuracy_score(y_test, y_test_pred_nn)
print(f"\nNeural Network Test Accuracy: {test_accuracy_nn:.2f}")
```

```
print("\nClassification Report (Neural Network on Test Data):")
print(classification_report(y_test, y_test_pred_nn))
```

```
Neural Network Test Accuracy: 0.90

Classification Report (Neural Network on Test Data):
              precision    recall  f1-score   support

           0       0.40      0.02      0.04      5006
           1       0.90      1.00      0.94     43179

    accuracy                           0.90     48185
   macro avg       0.65      0.51      0.49     48185
weighted avg       0.85      0.90      0.85     48185


C:\Users\taylo\anaconda3\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (10) reached and
the optimization hasn't converged yet.
  warnings.warn(
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

Cosine similarity

December 10, 2024

[1]:
```
import pandas as pd
import numpy as np
```

[159]:
```
# import the user data
df = pd.read_csv('user_rankings_v3_0-10000.csv', low_memory=False,␣
 ↪index_col='Unnamed: 0')
df2 = pd.read_csv('user_rankings_v3_10000-20000.csv', low_memory=False,␣
 ↪index_col='Unnamed: 0')
df3 = pd.read_csv('user_rankings_v3_20000-30000.csv', low_memory=False,␣
 ↪index_col='Unnamed: 0')
df2
```

[159]:

	user	score	title
0	Kabaki	10	Berserk
1	Kabaki	10	The Horizon
2	Kabaki	9	Three Days of Happiness
3	Kabaki	8	Death Note
4	Kabaki	8	Emanon: Memories of Emanon
…	…	…	…
1150294	maglevtrain	6	Genkai Dungeon no Hanshoku Jijou
1150295	maglevtrain	6	The Teen Web Novelist Is a Girl Magnet: Now My…
1150296	maglevtrain	6	Yuusha-sama, Sakuya mo Otanoshimi deshita ne.
1150297	maglevtrain	5	March Comes in Like a Lion
1150298	maglevtrain	5	Kanojo wa Kannou Shousetsuka

[1150299 rows x 3 columns]

[161]:
```
# merge the user data into one dataframe
user_data = pd.concat([df, df2, df3], ignore_index=True)
```

[167]:
```
#check the output and size
print(user_data.info())
user_data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3400016 entries, 0 to 3400015
Data columns (total 3 columns):
 #   Column  Dtype
```

```
---  ------  -----
 0   user    object
 1   score   object
 2   title   object
dtypes: object(3)
memory usage: 77.8+ MB
None
```

[167]:

	user	score	title
0	RakaanG	9	My Hero Academia: Vigilantes
1	RakaanG	9	My Quiet Blacksmith Life in Another World
2	RakaanG	8	My Hero Academia
3	RakaanG	8	Dr. Stone
4	RakaanG	8	Kujonin
…	…	…	…
3400011	StyxParadise	-	Douka Ore wo Houtteoite Kure: Nazeka Bocchi no…
3400012	StyxParadise	-	Chigau Miyahara Omae ja Nai!
3400013	StyxParadise	-	The Reincarnated Marriage of a Hero and Sage
3400014	StyxParadise	-	Kusunoki's Flunking Her High School Glow-Up
3400015	StyxParadise	-	Tune In to the Midnight Heart

[3400016 rows x 3 columns]

[169]:
```
# how many null values do we have?
user_data.isna().sum()
```

[169]:
```
user         0
score    45068
title    42225
dtype: int64
```

we don't want to use the missing data in this situation. If we were to assume someones score was the mean/median score value we might suggest books based on false data.

[172]:
```
# drop missing titles and scores
user_data.dropna(subset = ['title', 'score'], ignore_index = True, inplace=True)
user_data
```

[172]:

	user	score	title
0	RakaanG	9	My Hero Academia: Vigilantes
1	RakaanG	9	My Quiet Blacksmith Life in Another World
2	RakaanG	8	My Hero Academia
3	RakaanG	8	Dr. Stone
4	RakaanG	8	Kujonin
…	…	…	…
3346748	StyxParadise	-	Douka Ore wo Houtteoite Kure: Nazeka Bocchi no…
3346749	StyxParadise	-	Chigau Miyahara Omae ja Nai!
3346750	StyxParadise	-	The Reincarnated Marriage of a Hero and Sage

```
3346751  StyxParadise     -        Kusunoki's Flunking Her High School Glow-Up
3346752  StyxParadise     -                     Tune In to the Midnight Heart

[3346753 rows x 3 columns]
```

[174]:
```
# how many null values do we have after we dropped title and score?
user_data.isna().sum()
```

[174]:
```
user     0
score    0
title    0
dtype: int64
```

[176]:
```
# Whats the total count of each score?
user_data['score'].value_counts()
```

[176]:
```
score
-     1108106
8      601337
7      551843
9      358873
6      263447
10     259156
5      115494
4       45681
3       20193
1       11906
2       10717
Name: count, dtype: int64
```

8 is the most used when refering to the score of a book.
We can see that there are 1 Million un-scored books, labeld as '-'. These should be removed.
These books have not been read and rated. They are typically on there to keep track of books people want to read next.
Because these are missing scores linked to book titles filling them with a median score might negativly change the book the model recommends.

[180]:
```
# remove the '-' rows in user_data
user_data['score'] = pd.to_numeric(user_data['score'], errors='coerce',␣
 ↪downcast='integer')
# check to see if it was convereted
user_data['score'].value_counts(dropna=False)
```

[180]:
```
score
NaN     1108106
8.0      601337
7.0      551843
9.0      358873
```

```
6.0      263447
10.0     259156
5.0      115494
4.0       45681
3.0       20193
1.0       11906
2.0       10717
Name: count, dtype: int64
```

[182]:
```
user_data.dropna(subset='score', inplace=True)
# check to see if it was convereted
user_data['score'].value_counts(dropna=False)
```

[182]:
```
score
8.0     601337
7.0     551843
9.0     358873
6.0     263447
10.0    259156
5.0     115494
4.0      45681
3.0      20193
1.0      11906
2.0      10717
Name: count, dtype: int64
```

[184]:
```
high_scores = user_data[user_data['score'] >= 7]
top_20_manga = user_data.groupby(['title'])[['score']].count().
 ↪sort_values(by='score', ascending=False)
top_20_manga.head(20)
```

[184]:

title	score
Solo Leveling	10372
Chainsaw Man	9686
Berserk	8573
Demon Slayer: Kimetsu no Yaiba	7098
Attack on Titan	6901
Jujutsu Kaisen	6732
One Piece	6607
Tokyo Ghoul	6512
Horimiya	5997
One-Punch Man	5688
Kaguya-sama: Love Is War	5652
Spy x Family	5556
Goodnight Punpun	5502
My Hero Academia	5050

```
Bleach                                                  4877
[Oshi No Ko]                                            4679
A Silent Voice                                          4630
Naruto                                                  4580
I sold my life for ten thousand yen per year.           4484
Goodbye, Eri                                            4467
```

The score above is how many people have added the book to their list. By a metric of people intersted in the book solo leveling has the most people reading it. Chainsaw man which has become a popular anime is in second and Berserk which normally sits at the top of the most popular list is in third using this metric.

[192]:
```
# export the data to a csv
user_data.to_csv('cleaned_user_data_v2.csv')
```

1 Base Recommendation System

1.0.1 Cosine Similarity :

using cosine similarity we can use a simple method to produce very quick results.
This method might not give very interesting results but they will give popular book recomendations. If we want more variation we should try some other machine learning methods.

[187]:
```
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_score, recall_score, f1_score,␣
 ↪mean_absolute_error, mean_squared_error, confusion_matrix
import numpy as np

# Create the user-item interaction matrix
interaction_matrix = user_data.pivot_table(index='user', columns='title',␣
 ↪values='score', fill_value=0)

# Convert the interaction matrix to a numpy array (ensure it's numerical)
interaction_matrix_values = interaction_matrix.values

# Step 1: Train-test split
train_data, test_data = train_test_split(interaction_matrix_values, test_size=0.
 ↪2, random_state=42)


# Compute item-item similarity using cosine similarity
similarity_matrix = cosine_similarity(interaction_matrix.T)
similarity_df = pd.DataFrame(
    similarity_matrix, index=interaction_matrix.columns,␣
 ↪columns=interaction_matrix.columns
)
```

```
# Function to recommend books based on a given book title
def recommend_books(book_title, similarity_df, top_n=5):
    if book_title not in similarity_df.index:
        return f"{book_title} not found in the dataset."

    # Get the most similar books
    similar_books = similarity_df[book_title].sort_values(ascending=False).
 ↪iloc[1:top_n+1]
    return similar_books.index.tolist()
```

[190]:
```
# Example: Recommend books similar to "My Hero Academia"
user_favorite_book = "My Hero Academia"
recommendations = recommend_books(user_favorite_book, similarity_df)

recommendations
```

[190]:
```
['Jujutsu Kaisen',
 'Demon Slayer: Kimetsu no Yaiba',
 'One-Punch Man',
 'Attack on Titan',
 'One Piece']
```

[100]:
```
# Evaluate recommendations for test users
def evaluate_recommendations(test_data, n=5):
    y_true = []
    y_pred = []

    for user_index in range(test_data.shape[0]):
        # Get actual ratings from the test data for this user
        actual_ratings = test_data[user_index]

        # Identify books that were rated positively in the test set
        actual_books = np.where(actual_ratings > 0)[0]

        # Get the corresponding user ID from the interaction matrix
        user_id = interaction_matrix.index[user_index]

        # Generate recommendations for this user
        recommended_books = recommend_collaborative(user_id, n=n)

        # Convert recommended book titles to indices
        recommended_books_indices = [interaction_matrix.columns.get_loc(book)␣
 ↪for book in recommended_books if book in interaction_matrix.columns]

        # Compare recommendations with actual books (binary classification)
```

```
        y_true.extend([1 if i in actual_books else 0 for i in␣
 ↪recommended_books_indices])
        y_pred.extend([1] * len(recommended_books_indices))

    # Calculate precision and recall
    precision = precision_score(y_true, y_pred, zero_division=0)
    recall = recall_score(y_true, y_pred, zero_division=0)

    return precision, recall

# Run the evaluation
precision, recall = evaluate_recommendations(test_data, n=5)
print(f"Precision: {precision:.2f}, Recall: {recall:.2f}")
```

```
Precision: 0.23, Recall: 1.00
```

We end up with the same scores as with TruncakedSVD when we use only 10 n_components. The more components we use the lower the Precision becomes, but that is just because we are adding recommendations that the person might not have seen yet.

2 TruncatedSVD

[88]:
```
from sklearn.decomposition import TruncatedSVD
from sklearn.model_selection import train_test_split
import numpy as np

# Prepare the interaction matrix
interaction_matrix = user_data.pivot_table(index='user', columns='title',␣
 ↪values='score', fill_value=0)

# Convert the interaction matrix to a numpy array (ensure it's numerical)
interaction_matrix_values = interaction_matrix.values

# Step 1: Train-test split
train_data, test_data = train_test_split(interaction_matrix_values, test_size=0.
 ↪2, random_state=42)

# Step 2: Apply Singular Value Decomposition (SVD) on training data for␣
 ↪collaborative filtering
svd = TruncatedSVD(n_components=10, random_state=42)
latent_matrix_train = svd.fit_transform(train_data)
latent_matrix_item = svd.components_

# Step 3: Function to recommend books using collaborative filtering
def recommend_collaborative(user_id, n=5):
    if user_id not in interaction_matrix.index:
        return ["No user data available for recommendations."]
```

```
    # Get the user's interaction vector from the training latent matrix
    user_index = interaction_matrix.index.get_loc(user_id)
    if user_index >= latent_matrix_train.shape[0]:  # Handle cases where user␣
 ↪may not be in the training set
        return ["User not in the training set for recommendations."]

    user_vector = latent_matrix_train[user_index]

    # Calculate similarity scores with all items
    scores = np.dot(user_vector, latent_matrix_item)

    # Rank books based on scores
    recommended_books = np.argsort(scores)[::-1][:n]
    recommended_book_titles = interaction_matrix.columns[recommended_books]

    return recommended_book_titles.tolist()

# Test collaborative filtering recommendations for a sample user
sample_user_id = interaction_matrix.index[1]  # Replace with a specific user ID␣
 ↪if needed
collab_recommendations = recommend_collaborative(sample_user_id, n=5)

collab_recommendations
```

[88]:
```
['Berserk',
 'Chainsaw Man',
 'Solo Leveling',
 'My Dress-Up Darling',
 "Don't Toy With Me, Miss Nagatoro"]
```

[121]:
```
# Test collaborative filtering recommendations for a sample user
sample_user_id = interaction_matrix.index[0]  # Replace with a specific user ID␣
 ↪if needed
collab_recommendations = recommend_collaborative(sample_user_id, n=5)

collab_recommendations
```

[121]:
```
['Chainsaw Man', 'Goodbye, Eri', 'Goodnight Punpun', 'Look Back', 'Fire Punch']
```

[123]:
```
user_data[user_data['user'] == sample_user_id]
```

[123]:

	user	score	title
692294	--AOD--	9.0	Initial D
692295	--AOD--	9.0	Pokémon Adventures
692296	--AOD--	9.0	Kami nomi zo Shiru Sekai
692297	--AOD--	9.0	One-Punch Man

```
692298  --AOD--    9.0                                                Bastard
692299  --AOD--    9.0                                             Sweet Home
692300  --AOD--    9.0                                              Weak Hero
692301  --AOD--    8.0                                  Girls of the Wild's
692302  --AOD--    8.0                      A Dating Sim of Life or Death
692303  --AOD--    8.0                                              Lookism
692304  --AOD--    8.0                                         Super Secret
692305  --AOD--    8.0                       Please Go Home, Miss Akutsu!
692306  --AOD--    8.0                                         Tower of God
692307  --AOD--    8.0                                          Shotgun Boy
692308  --AOD--    8.0                                            The Gamer
692309  --AOD--    8.0                                             Oh! Holy
692310  --AOD--    8.0                                                 Hive
692311  --AOD--    7.0                                              B.O.D.Y.
692312  --AOD--    7.0                                    Good Morning Call
692313  --AOD--    7.0                               Pokémon Gold & Silver
692314  --AOD--    7.0                                  See Me After Class
692315  --AOD--    7.0                               Classroom of the Elite
692316  --AOD--    7.0  Trapped in a Dating Sim: The World of Otome Ga…
692317  --AOD--    7.0                               Girlfriend, Girlfriend
692318  --AOD--    7.0                  I Don't Know If It's Love or Magic!
692319  --AOD--    7.0                                         Random Chat
692320  --AOD--    7.0                          +99 Reinforced Wooden Stick
692321  --AOD--    7.0  Love Comedy Manga ni Haitteshimatta node, Oshi…
692322  --AOD--    7.0  Netorare Manga no Kuzu Otoko ni Tensei shita H…
692323  --AOD--    7.0                                       Just One Bite!
692324  --AOD--    6.0                             Oh, Those Hanazono Twins
```

Just doing a manual check on these recommendations and it looks very accurate for the books they are reading.

[130]:
```python
# Step 4: Evaluate recommendations for test users
def evaluate_recommendations(test_data, n=5):
    y_true = []
    y_pred = []

    for user_index in range(test_data.shape[0]):
        # Get actual ratings from the test data for this user
        actual_ratings = test_data[user_index]

        # Identify books that were rated positively in the test set
        actual_books = np.where(actual_ratings > 4)[0]

        # Get the corresponding user ID from the interaction matrix
        user_id = interaction_matrix.index[user_index]

        # Generate recommendations for this user
```

```
        recommended_books = recommend_collaborative(user_id, n=n)

        # Convert recommended book titles to indices
        recommended_books_indices = [interaction_matrix.columns.get_loc(book)␣
↪for book in recommended_books if book in interaction_matrix.columns]

        # Compare recommendations with actual books (binary classification)
        y_true.extend([1 if i in actual_books else 0 for i in␣
↪recommended_books_indices])
        y_pred.extend([1] * len(recommended_books_indices))

    # Calculate precision and recall
    precision = precision_score(y_true, y_pred, zero_division=0)
    recall = recall_score(y_true, y_pred, zero_division=0)

    return precision, recall

# Step 5: Run the evaluation
precision, recall = evaluate_recommendations(test_data, n=5)
print(f"Precision: {precision:.2f}, Recall: {recall:.2f}")
```

```
Precision: 0.23, Recall: 1.00
```

A recall score shows that the recommendation system is recommending 100% of the books the users have already read. which is good that means that its able to predict what books an individual might want to read. A low Precision score in this case is not a bad sign. Its saying that even though we recommended all the books the individual has read we are also recommeding books that they have not. We will have to further test this data on users.

3 Surprise Model -

The Surprise model will take the dataset of users, items (e.g., titles), and ratings to learn user preferences using collaborative filtering.

[103]:
```
import pandas as pd
from surprise import Dataset, Reader, SVD
from surprise.model_selection import train_test_split
from surprise import accuracy

# Load the dataset
file_path = 'cleaned_user_data_v1.csv'
df = pd.read_csv(file_path)

# Prepare data for Surprise library
reader = Reader(rating_scale=(0, 10))
data = Dataset.load_from_df(df[['user', 'title', 'score']], reader)

# Train-test split
```

```
trainset, testset = train_test_split(data, test_size=0.2, random_state=42)

# Apply SVD for matrix factorization
svd = SVD()
svd.fit(trainset)

# Make predictions on the test set
predictions = svd.test(testset)

# Calculate RMSE for evaluation
rmse = accuracy.rmse(predictions)

# Calculate MAE for evaluation
mae = accuracy.mae(predictions)

# Calculate FCP for evaluation
fcp = accuracy.fcp(predictions)
```

```
RMSE: 1.1651
MAE:  0.8584
FCP:  0.6753
```

An RMSE of 1.16 is a great improvement over the baseline of 2.2. It is normal for such a sparce dataset to have a RMSE between 1-1.5
An MAE of .85 is a great improvement over the baseline model below.
The FCP calculates how the model orders the recommendations based on the rankings. Its not perfect at how it organizes the manga recommendations.

Once we test on real people we can see how the models perform. When we are only recommending 5 books the rankings might not matter as much. Individuals will typically look into each book vs just picking the top recommendation.

4 RMSE Baseline

[78]:
```
from surprise import NormalPredictor

# Apply NormalPredictor as the baseline model
baseline_model = NormalPredictor()
baseline_model.fit(trainset)

# Make predictions on the test set
baseline_predictions = baseline_model.test(testset)

# Calculate RMSE for evaluation
baseline_rmse = accuracy.rmse(baseline_predictions)

# Calculate MAE for evaluation
mae = accuracy.mae(baseline_predictions)
```

```
# Calculate FCP for evaluation
fcp = accuracy.fcp(baseline_predictions)
```

```
RMSE: 2.2008
MAE:  1.7401
FCP:  0.4991
```

We can see that we get more accurate predictions from the surprise model.
While these are not perfect predictions this will give us room to provide a possibly better range of recommendations.

5 Hyperparameter Tuning

[109]:
```
from surprise import SVD, Dataset, Reader
from surprise.model_selection import GridSearchCV

# Load the dataset
file_path = 'cleaned_user_data_v1.csv'
df = pd.read_csv(file_path)
reader = Reader(rating_scale=(0, 10))
data = Dataset.load_from_df(df[['user', 'title', 'score']], reader)

# Define the parameter grid
param_grid = {
    'n_factors': [20, 50, 100],
    'lr_all': [0.002, 0.005, 0.01],
    'reg_all': [0.02, 0.1, 0.2]
}

# Set up GridSearchCV to find the best hyperparameters
gs = GridSearchCV(SVD, param_grid, measures=['rmse'], cv=5, n_jobs=-1)

# Fit GridSearchCV
gs.fit(data)

# Extract and display the best RMSE score and best parameters
print(f"Best RMSE score: {gs.best_score['rmse']}")
print(f"Best parameters: {gs.best_params['rmse']}")
```

```
Best RMSE score: 1.1367838738399716
Best parameters: {'n_factors': 100, 'lr_all': 0.01, 'reg_all': 0.1}
```

[199]:
```
from surprise import Dataset, Reader, SVD

# Load the dataset
file_path = 'cleaned_user_data_v1.csv'
df = pd.read_csv(file_path)
```

```
data_cleaned = df[['user', 'title', 'score']]
# Prepare data for Surprise library
reader = Reader(rating_scale=(0, 10))
data = Dataset.load_from_df(df[['user', 'title', 'score']], reader)

# Train-test split and train the model
trainset = data.build_full_trainset()
model = SVD()
model.fit(trainset)

# Define a function to recommend manga based on user input
def get_recommendations(user_input, existing_data, model, reader):
    # Assign a new user ID for this input session
    user_id = "new_user"

    # Create a DataFrame from the user input
    user_ratings = pd.DataFrame({
        "user": [user_id] * len(user_input),
        "title": [item[0] for item in user_input],
        "score": [item[1] for item in user_input],
    })

    # Merge user ratings with the existing dataset
    updated_df = pd.concat([existing_data, user_ratings], ignore_index=True)

    # Reload the dataset with the new data
    updated_data = Dataset.load_from_df(updated_df, reader)
    updated_trainset = updated_data.build_full_trainset()
    model.fit(updated_trainset)

    # Predict scores for all titles not rated by the user
    all_titles = set(existing_data['title'])
    rated_titles = set(user_ratings['title'])
    unrated_titles = all_titles - rated_titles

    predictions = [
        (title, model.predict(user_id, title).est) for title in unrated_titles
    ]

    # Sort by estimated score and recommend the top 5
    recommendations = sorted(predictions, key=lambda x: x[1], reverse=True)[:5]
    return recommendations

# Example user input
user_input = [
    ("Fruits Basket", 8),
    ("Sono Bisque Doll wa Koi wo Suru", 9),
```

```
    ("Kaichou wa Maid-sama!", 7)
]

# Get recommendations
recommendations = get_recommendations(user_input, data_cleaned, model, reader)

# Display recommendations
recommendations_df = pd.DataFrame(recommendations, columns=["Title", "Estimated␣
↪Score"])
```

[201]:
```
# Example user input
user_input = [
    ("Fruits Basket", 8),
    ("Horimiya", 9),
    ("Kaichou wa Maid-sama!", 7)
]

# Get recommendations
recommendations = get_recommendations(user_input, data_cleaned, model, reader)

# Display recommendations
recommendations_df = pd.DataFrame(recommendations, columns=["Title", "Estimated␣
↪Score"])
recommendations_df
```

[201]:

	Title	Estimated Score
0	Berserk	9.601225
1	The House in Fata Morgana	9.548774
2	JoJo's Bizarre Adventure Part 7: Steel Ball Run	9.541119
3	BERSERK	9.515127
4	Monster	9.428304

This is just a slight improvement over the Surprise Model. Not something that is significant enough to increase compute by using it.

SVD-NN Model

December 10, 2024

1 SVD + Nearest Neighbors

We are using SVD in order to reduce the dimensions to help process the model in a timely manner. There is a chance it can negativly impact the model performance if we lower the n_components too much.

[2]:
```
import pandas as pd
import numpy as np
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.decomposition import TruncatedSVD
from sklearn.neighbors import NearestNeighbors
from scipy.sparse import csr_matrix
import joblib  # To save and load the model

# Assuming user_data is the dataframe containing user interactions with columns:
 ↪ 'user', 'title', 'score'
user_data = pd.read_csv('cleaned_user_data_v1.csv')

# Create the user-item interaction matrix and limit it to the first 10,000 users
interaction_matrix = user_data.pivot_table(index='user', columns='title',␣
 ↪values='score', fill_value=0)



# # Save the interaction matrix to a pkl file
# joblib.dump(interaction_matrix, 'interaction_matrix_20k.pkl')


# Convert the interaction matrix to a sparse matrix
interaction_sparse = csr_matrix(interaction_matrix.values)

# Reduce dimensionality with Truncated SVD
svd = TruncatedSVD(n_components=50, random_state=42)
reduced_interaction_matrix = svd.fit_transform(interaction_sparse)

# Fit the Nearest Neighbors model on the reduced user-item matrix
nn_model = NearestNeighbors(metric='cosine', algorithm='auto', n_neighbors=10,␣
 ↪n_jobs=-1)
```

```
nn_model.fit(reduced_interaction_matrix)

# # Save the Truncated SVD transformer and Nearest Neighbors model to pkl files
# joblib.dump(svd, 'svd_model_20.pkl')
# joblib.dump(nn_model, 'nn_model_20.pkl')
```

[2]: ['nn_model.pkl']

2 Model testing

[4]:
```
import pandas as pd
import numpy as np
import joblib  # To load the saved model
from sklearn.decomposition import TruncatedSVD

# Load the saved interaction matrix, Truncated SVD transformer, and Nearest␣
 ↪Neighbors model
interaction_matrix = joblib.load('interaction_matrix_30k.pkl')
svd = joblib.load('svd_model_30k.pkl')
nn_model = joblib.load('nn_model_30k.pkl')

# Function to recommend books based on similar users using the loaded models
def recommend_books_for_new_user(new_user_ratings, nn_model,␣
 ↪interaction_matrix, svd, top_n=5):
    # Add the new user's ratings to the interaction matrix
    new_user_df = pd.DataFrame(new_user_ratings, index=['new_user'],␣
 ↪columns=interaction_matrix.columns).fillna(0)

    # Reduce the new user's ratings to the reduced space using the SVD model
    new_user_vector = svd.transform(new_user_df.values)  # Shape: (1,␣
 ↪n_components)

    # Find the top similar users using the Nearest Neighbors model
    distances, indices = nn_model.kneighbors(new_user_vector, n_neighbors=10)
    similar_users = interaction_matrix.index[indices.flatten()]

    # Aggregate the ratings of similar users weighted by their similarity to␣
 ↪the new user
    weighted_sum = np.zeros(interaction_matrix.shape[1])
    similarity_sum = 0

    for idx, user in enumerate(similar_users):
        user_similarity_score = 1 - distances.flatten()[idx]  # Similarity = 1␣
 ↪- distance
        weighted_sum += interaction_matrix.loc[user].values *␣
 ↪user_similarity_score
```

```
            similarity_sum += user_similarity_score

        # Avoid division by zero
        if similarity_sum != 0:
            weighted_sum /= similarity_sum

        # Create a list of recommended books by ranking the highest scores not yet␣
 ↪rated by the new user
        new_user_rated = new_user_df.loc['new_user']
        recommendations = pd.Series(weighted_sum, index=interaction_matrix.columns)

        # Remove books already rated by the new user
        recommendations = recommendations[new_user_rated == 0]

        # Return the top N recommendations
        top_recommendations = recommendations.sort_values(ascending=False).
 ↪head(top_n)

        return top_recommendations.index.tolist()

# Example usage for a new user input:
new_user_ratings = {
    'Horimiya': 8,
    'Fruits Basket': 8,
    'Kaichou wa Maid-sama!': 7
}

# Generate recommendations using the loaded models
recommended_books = recommend_books_for_new_user(new_user_ratings, nn_model,␣
 ↪interaction_matrix, svd, top_n=5)
print("Recommended Books for You:", recommended_books)
```

```
Recommended Books for You: ['Cigarette & Cherry', 'No Longer Heroine', 'Gakuen
Alice', 'Love Me, Love Me Not', 'Happy Marriage?!']
```

Doing a manual evaluation on what this is recommedning its doing a good job giving similar genres even though we dont know what genre each manga is. It is also giving manga with similar ratings which makes sense becasue its clustering users preferences together. So if the input user rates books 8, 8, 7 then it will match them with others that rated books the same.

Another way to evaluate the model is to let users interact with it and see if they choose any of the recommendations. Spotify does this through click tracking and if someone plays a song after they recommended it. This would be hard to do in this case because we dont have an app to collect and track if someone read a manga or not. What we can do is add a button on our website to ask if the recommendations were helpful or not.

3 Statistical Metrics

[2]:
```
import pandas as pd
import numpy as np
from sklearn.decomposition import TruncatedSVD
from sklearn.neighbors import NearestNeighbors
from scipy.sparse import csr_matrix
import joblib
from sklearn.metrics import mean_absolute_error, mean_squared_error
from math import sqrt
import random

# Load user interaction data
user_data = pd.read_csv('cleaned_user_data_v1.csv')

# Create user-item interaction matrix and limit it to the first 10,000 users
interaction_matrix = user_data.pivot_table(index='user', columns='title',␣
 ↪values='score', fill_value=0)


# Create train-test split (leave out 20% of ratings for testing)
interaction_matrix_train = interaction_matrix.copy()
test_set = []

random.seed(42)
for user in interaction_matrix.index:
    rated_books = interaction_matrix.loc[user].values.nonzero()[0]
    n_ratings = len(rated_books)
    if n_ratings > 1:
        n_test = max(1, n_ratings // 5)  # Leave out 20% of ratings
        test_indices = random.sample(list(rated_books), n_test)
        for idx in test_indices:
            interaction_matrix_train.at[user, interaction_matrix.columns[idx]]␣
 ↪= 0
            test_set.append((user, interaction_matrix.columns[idx],␣
 ↪interaction_matrix.loc[user, interaction_matrix.columns[idx]]))

# **Sample the Test Set** to make metrics calculation faster
sample_test_set = random.sample(test_set, min(5000, len(test_set)))  # Sample␣
 ↪1000 user-item pairs for evaluation

# Save the training interaction matrix
joblib.dump(interaction_matrix_train, 'interaction_matrix_train.pkl')

# Convert interaction matrix to sparse matrix
interaction_sparse = csr_matrix(interaction_matrix_train.values)
```

```
# Train the SVD and Nearest Neighbors models
svd = TruncatedSVD(n_components=50, random_state=42)
reduced_interaction_matrix = svd.fit_transform(interaction_sparse)
nn_model = NearestNeighbors(metric='cosine', algorithm='auto', n_neighbors=10,␣
 ↪n_jobs=-1)
nn_model.fit(reduced_interaction_matrix)

# Save the models
joblib.dump(svd, 'svd_model_train.pkl')
joblib.dump(nn_model, 'nn_model_train.pkl')
```

[2]:
```
['nn_model_train.pkl']
```

[5]:
```
# Load the saved models
interaction_matrix_train = joblib.load('interaction_matrix_train.pkl')
svd = joblib.load('svd_model_train.pkl')
nn_model = joblib.load('nn_model_train.pkl')

# Function to predict ratings
def predict_rating(user, item, interaction_matrix, svd, nn_model):
    # Check if user or item is present in training data
    if user not in interaction_matrix.index or item not in interaction_matrix.
 ↪columns:
        return None  # User or item not in training set

    # Reduce the user's existing ratings using SVD
    user_vector = svd.transform(interaction_matrix.loc[user].values.reshape(1,␣
 ↪-1))

    # Find similar users using the reduced user vector
    distances, indices = nn_model.kneighbors(user_vector, n_neighbors=5)  # Use␣
 ↪5 nearest neighbors
    similar_users = interaction_matrix.index[indices.flatten()]

    # Aggregate ratings to predict rating for the item
    weighted_sum = 0
    similarity_sum = 0
    for idx, sim_user in enumerate(similar_users):
        user_similarity_score = 1 - distances.flatten()[idx]
        if user_similarity_score > 0:
            sim_user_rating = interaction_matrix.at[sim_user, item]
            weighted_sum += sim_user_rating * user_similarity_score
            similarity_sum += user_similarity_score

    # Return predicted rating if we have a valid similarity sum
    if similarity_sum > 0:
        return weighted_sum / similarity_sum
```

```
    else:
        return interaction_matrix[item].mean()  # Default to mean rating if no similar users

# Predict for the sample test set and calculate metrics
y_true = []
y_pred = []

for (user, item, actual_rating) in sample_test_set:
    predicted_rating = predict_rating(user, item, interaction_matrix_train, svd, nn_model)
    if predicted_rating is not None:
        y_true.append(actual_rating)
        y_pred.append(predicted_rating)

# Calculate MAE and RMSE
mae = mean_absolute_error(y_true, y_pred)
rmse = sqrt(mean_squared_error(y_true, y_pred))

print(f"Mean Absolute Error (MAE): {mae}")
print(f"Root Mean Squared Error (RMSE): {rmse}")

# Calculate Precision and Recall (Top-N Recommendations)
def recommend_top_n(user, interaction_matrix, svd, nn_model, top_n=5):
    # Reduce the user's existing ratings using SVD
    user_vector = svd.transform(interaction_matrix.loc[user].values.reshape(1, -1))

    # Find similar users and aggregate ratings to generate top-N recommendations
    distances, indices = nn_model.kneighbors(user_vector, n_neighbors=5)
    similar_users = interaction_matrix.index[indices.flatten()]

    # Aggregate ratings from similar users
    weighted_sum = np.zeros(interaction_matrix.shape[1])
    similarity_sum = 0
    for idx, sim_user in enumerate(similar_users):
        user_similarity_score = 1 - distances.flatten()[idx]
        weighted_sum += interaction_matrix.loc[sim_user].values * user_similarity_score
        similarity_sum += user_similarity_score

    if similarity_sum > 0:
        weighted_sum /= similarity_sum

    recommendations = pd.Series(weighted_sum, index=interaction_matrix.columns)
    return recommendations.sort_values(ascending=False).head(top_n).index.tolist()
```

```
# Evaluate Precision and Recall for the sample test set
precision_sum = 0
recall_sum = 0
top_n = 5

for user, item, _ in sample_test_set:
    if user in interaction_matrix_train.index:
        recommended_items = recommend_top_n(user, interaction_matrix_train,␣
 ↪svd, nn_model, top_n=top_n)
        if item in recommended_items:
            precision_sum += 1
            recall_sum += 1

precision = precision_sum / (len(sample_test_set) * top_n)
recall = recall_sum / len(sample_test_set)

print(f"Precision@{top_n}: {precision}")
print(f"Recall@{top_n}: {recall}")
```

```
Mean Absolute Error (MAE): 6.6785222224574685
Root Mean Squared Error (RMSE): 6.951781746429006
Precision@5: 0.00052
Recall@5: 0.0026
```

The metrics above were only ran on 10000 user item pairs. Even though the recommendations from the whole model look like good recommendation this shows that it is poor at predicting what someone might score a manga. An MAE of 6.6 is pretty bad when the scale goes to 10.

Potential Problems and Solutions: The metrics are suggesting that the model may have a low quality of recommendations and inaccurate predictions. Let's consider some common issues and approaches to improve the performance:

1. Cold Start Problem: Users and items in the test set may not have enough data to make accurate recommendations. Solution: Consider hybrid approaches where content-based filtering is combined with collaborative filtering to generate recommendations for items or users with few interactions.
2. High Sparsity: In recommendation systems, sparsity is common, where users rate only a few items out of a large catalog. The performance of collaborative filtering can degrade significantly with sparse data. Solution: Apply dimensionality reduction techniques, such as SVD (which I am using already), with a higher number of components (n_components) to improve the ability to capture user-item relationships. Consider Matrix Factorization using more advanced techniques like Alternating Least Squares (ALS) to better handle sparsity.
3. Model Complexity: Using only 50 components in the SVD might not be enough to capture the relationships between users and items accurately. Solution: Increase the number of components in SVD to 100 or 200 to allow the model to learn more latent features. Be cautious about the impact on computation, but this may improve the model's representation power.
4. Nearest Neighbors Selection: Reducing n_neighbors to 5 made the prediction process faster, but it may have also reduced the accuracy by limiting the user pool used to make predictions.

Solution: Experiment with a higher number of neighbors (n_neighbors=10 or 15) and see if it positively impacts the prediction quality without significantly compromising computation time.

3.0.1 Added Extra n_components to test if it will produce better results

[2]:
```
import pandas as pd
import numpy as np
from sklearn.decomposition import TruncatedSVD
from sklearn.neighbors import NearestNeighbors
from scipy.sparse import csr_matrix
import joblib
from sklearn.metrics import mean_absolute_error, mean_squared_error
from math import sqrt
import random

# Load user interaction data
user_data = pd.read_csv('cleaned_user_data_v1.csv')

# Create user-item interaction matrix and limit it to the first 10,000 users
interaction_matrix = user_data.pivot_table(index='user', columns='title',␣
 ↪values='score', fill_value=0)
interaction_matrix = interaction_matrix.iloc[:10000, :]

# Create train-test split (leave out 20% of ratings for testing)
interaction_matrix_train = interaction_matrix.copy()
test_set = []

random.seed(42)
for user in interaction_matrix.index:
    rated_books = interaction_matrix.loc[user].values.nonzero()[0]
    n_ratings = len(rated_books)
    if n_ratings > 1:
        n_test = max(1, n_ratings // 5)  # Leave out 20% of ratings
        test_indices = random.sample(list(rated_books), n_test)
        for idx in test_indices:
            interaction_matrix_train.at[user, interaction_matrix.columns[idx]]␣
 ↪= 0
            test_set.append((user, interaction_matrix.columns[idx],␣
 ↪interaction_matrix.loc[user, interaction_matrix.columns[idx]]))

# **Sample the Test Set** to make metrics calculation faster
sample_test_set = random.sample(test_set, min(5000, len(test_set)))  # Sample␣
 ↪1000 user-item pairs for evaluation

# # Save the training interaction matrix
# joblib.dump(interaction_matrix_train, 'interaction_matrix_train_20.pkl')
```

```
# Convert interaction matrix to sparse matrix
interaction_sparse = csr_matrix(interaction_matrix_train.values)

# Train the SVD and Nearest Neighbors models
svd = TruncatedSVD(n_components=20, random_state=42)
reduced_interaction_matrix = svd.fit_transform(interaction_sparse)
nn_model = NearestNeighbors(metric='cosine', algorithm='auto', n_neighbors=10,␣
 ↪n_jobs=-1)
nn_model.fit(reduced_interaction_matrix)

# # Save the models
# joblib.dump(svd, 'svd_model_train_20.pkl')
# joblib.dump(nn_model, 'nn_model_train_20.pkl')
```

[2]:
```
['nn_model_train_20.pkl']
```

[4]:
```
# Load the saved models
interaction_matrix_train = joblib.load('interaction_matrix_train_20.pkl')
svd = joblib.load('svd_model_train_20.pkl')
nn_model = joblib.load('nn_model_train_20.pkl')

# Function to predict ratings
def predict_rating(user, item, interaction_matrix, svd, nn_model):
    # Check if user or item is present in training data
    if user not in interaction_matrix.index or item not in interaction_matrix.
 ↪columns:
        return None  # User or item not in training set

    # Reduce the user's existing ratings using SVD
    user_vector = svd.transform(interaction_matrix.loc[user].values.reshape(1,␣
 ↪-1))

    # Find similar users using the reduced user vector
    distances, indices = nn_model.kneighbors(user_vector, n_neighbors=5)  # Use␣
 ↪5 nearest neighbors
    similar_users = interaction_matrix.index[indices.flatten()]

    # Aggregate ratings to predict rating for the item
    weighted_sum = 0
    similarity_sum = 0
    for idx, sim_user in enumerate(similar_users):
        user_similarity_score = 1 - distances.flatten()[idx]
        if user_similarity_score > 0:
            sim_user_rating = interaction_matrix.at[sim_user, item]
            weighted_sum += sim_user_rating * user_similarity_score
            similarity_sum += user_similarity_score
```

```
    # Return predicted rating if we have a valid similarity sum
    if similarity_sum > 0:
        return weighted_sum / similarity_sum
    else:
        return interaction_matrix[item].mean()  # Default to mean rating if no␣
 ↪similar users

# Predict for the sample test set and calculate metrics
y_true = []
y_pred = []

for (user, item, actual_rating) in sample_test_set:
    predicted_rating = predict_rating(user, item, interaction_matrix_train,␣
 ↪svd, nn_model)
    if predicted_rating is not None:
        y_true.append(actual_rating)
        y_pred.append(predicted_rating)

# Calculate MAE and RMSE
mae = mean_absolute_error(y_true, y_pred)
rmse = sqrt(mean_squared_error(y_true, y_pred))

print(f"Mean Absolute Error (MAE): {mae}")
print(f"Root Mean Squared Error (RMSE): {rmse}")

# Calculate Precision and Recall (Top-N Recommendations)
def recommend_top_n(user, interaction_matrix, svd, nn_model, top_n=5):
    # Reduce the user's existing ratings using SVD
    user_vector = svd.transform(interaction_matrix.loc[user].values.reshape(1,␣
 ↪-1))

    # Find similar users and aggregate ratings to generate top-N recommendations
    distances, indices = nn_model.kneighbors(user_vector, n_neighbors=5)
    similar_users = interaction_matrix.index[indices.flatten()]

    # Aggregate ratings from similar users
    weighted_sum = np.zeros(interaction_matrix.shape[1])
    similarity_sum = 0
    for idx, sim_user in enumerate(similar_users):
        user_similarity_score = 1 - distances.flatten()[idx]
        weighted_sum += interaction_matrix.loc[sim_user].values *␣
 ↪user_similarity_score
        similarity_sum += user_similarity_score

    if similarity_sum > 0:
        weighted_sum /= similarity_sum
```

```
    recommendations = pd.Series(weighted_sum, index=interaction_matrix.columns)
    return recommendations.sort_values(ascending=False).head(top_n).index.
 ↪tolist()

# Evaluate Precision and Recall for the sample test set
precision_sum = 0
recall_sum = 0
top_n = 5

for user, item, _ in sample_test_set:
    if user in interaction_matrix_train.index:
        recommended_items = recommend_top_n(user, interaction_matrix_train,␣
 ↪svd, nn_model, top_n=top_n)
        if item in recommended_items:
            precision_sum += 1
            recall_sum += 1

precision = precision_sum / (len(sample_test_set) * top_n)
recall = recall_sum / len(sample_test_set)

print(f"Precision@{top_n}: {precision}")
print(f"Recall@{top_n}: {recall}")
```

```
Mean Absolute Error (MAE): 6.718571765387836
Root Mean Squared Error (RMSE): 6.996115838563513
Precision@5: 0.00132
Recall@5: 0.0066
```

4 ALS model using Implicit library

[2]:
```
import pandas as pd
import numpy as np
from threadpoolctl import threadpool_limits
from scipy.sparse import csr_matrix
import joblib
from sklearn.metrics import mean_absolute_error, mean_squared_error
from math import sqrt
import random

threadpool_limits(1, "blas")

# Load user interaction data
user_data = pd.read_csv('cleaned_user_data_v1.csv')


# Create user-item interaction matrix and limit it to the first 10,000 users
```

```
interaction_matrix = user_data.pivot_table(index='user', columns='title',␣
 ↪values='score', fill_value=0)


# Create train-test split (leave out 20% of ratings for testing)
interaction_matrix_train = interaction_matrix.copy()
test_set = []

random.seed(42)
for user in interaction_matrix.index:
    rated_books = interaction_matrix.loc[user].values.nonzero()[0]  # Convert␣
 ↪to NumPy array and then get non-zero indices
    n_ratings = len(rated_books)
    if n_ratings > 1:
        n_test = max(1, n_ratings // 5)  # Leave out 20% of ratings
        test_indices = random.sample(list(rated_books), n_test)
        for idx in test_indices:
            interaction_matrix_train.at[user, interaction_matrix.columns[idx]]␣
 ↪= 0
            test_set.append((user, interaction_matrix.columns[idx],␣
 ↪interaction_matrix.loc[user, interaction_matrix.columns[idx]]))

# Save the training interaction matrix
#joblib.dump(interaction_matrix_train, 'interaction_matrix_train_10k.pkl')

# Convert interaction matrix to a sparse matrix in CSR format
interaction_sparse = csr_matrix(interaction_matrix_train.values)
interaction_sparse = interaction_sparse.tocsr()  # Explicitly ensure CSR format
```

[6]:
```
from implicit.als import AlternatingLeastSquares

# ALS model configuration and training
als_model = AlternatingLeastSquares(factors=100, regularization=0.1,␣
 ↪iterations=20, random_state=42)

# Implicit ALS expects the data in the form where items are rows and users are␣
 ↪columns, so transpose the matrix
als_model.fit(interaction_sparse.T.tocsr())

# Save the ALS model to pkl file
joblib.dump(als_model, 'als_model.pkl')
```

```
  0%|          | 0/20 [00:00<?, ?it/s]
```

[6]:
```
['als_model.pkl']
```

[18]:
```
import numpy as np
from scipy.sparse import csr_matrix
import pandas as pd
import joblib

# Load ALS model and interaction matrix
als_model = joblib.load('als_model.pkl')


# Get the list of manga titles from the original interaction matrix columns
manga_titles = interaction_matrix_train.columns

# Function to take user ratings and provide recommendations
def get_recommendations(user_ratings, als_model, manga_titles,␣
 ↪num_recommendations=10):
    """
    Get manga recommendations based on user input ratings.

    Parameters:
    - user_ratings: List of tuples (manga_title, rating), e.g., [("Manga A",␣
 ↪8), ("Manga B", 10)]
    - als_model: Trained ALS model
    - manga_titles: List of all manga titles
    - num_recommendations: Number of recommendations to return

    Returns:
    - List of recommended manga titles
    """
    # Create a user vector with all zero ratings initially
    user_vector = np.zeros(len(manga_titles))

    # Update the user vector with provided ratings
    for manga_title, rating in user_ratings:
        if manga_title in manga_titles:
            idx = manga_titles.get_loc(manga_title)  # Get index of manga title
            user_vector[idx] = rating

    # Convert user vector to CSR format for compatibility with ALS model
    user_sparse_vector = csr_matrix(user_vector)

    # Get user factors (latent factors for the new user)
    user_factors = als_model.user_factors.T @ user_sparse_vector.T

    # Compute scores for all items by multiplying item_factors by user_factors
    scores = als_model.item_factors @ user_factors.flatten()
```

```
    # Get the indices of the top N recommendations, excluding items already␣
 ↪rated by the user
    already_rated = set(np.nonzero(user_vector)[0])
    recommendations = [
        idx for idx in np.argsort(-scores) if idx not in already_rated
    ][:num_recommendations]

    # Get the recommended manga titles
    recommended_titles = [manga_titles[idx] for idx in recommendations]

    return recommended_titles


# Example user input: 3 manga titles with ratings
user_ratings_input = [
    ("Naruto", 8),
    ("Attack on Titan", 9),
    ("One Piece", 7)
]

# Get recommendations
recommendations = get_recommendations(user_ratings_input, als_model,␣
 ↪manga_titles)

# Print recommendations
print("Recommended Manga Titles:")
for title in recommendations:
    print(title)
```

```
Recommended Manga Titles:
Haru Ranman!
Judge
Days of Cool Idols
-6mm no Taboo
Futari no Himitsu
Come Rain or Shine
Hakushaku to Yobareta Otoko
"Nisekoi"
Kimi no Sei
Doku wo Kurawaba Sara made
```

~\Desktop\Portfolio-Project-Recommendation-System\final project folder\application.py

```python
# save this as app.py
from flask import Flask, request, render_template
import pandas as pd
import numpy as np
import joblib




# ---------------------------------------------------------------------------------------------------------------------#
# ------------------------------------------------ Flask Setup ---------------------------------------------------------#
# ---------------------------------------------------------------------asdf--------------------------------------------#
application = Flask(__name__)

# ---------------------------------------------------------------------------------------------------------------------#
# ------------------------------------------- Load Data and Models ---------------------------------------------------#
# ---------------------------------------------------------------------------------------------------------------------#
# Load the dataset of manga titles and links
manga_links_df = pd.read_csv('title_link_extra.csv')
manga_image_links_df = pd.read_csv('title-link-image-score-eng-title.csv')

# Load dataset containing manga and genres
manga_data = pd.read_csv('manga_list_3.csv')
manga_data['Title'] = manga_data['Link'].apply(lambda x: x.split('/')[-1].replace('_', ' '))

# Load the saved interaction matrix and models
interaction_matrix = joblib.load('interaction_matrix_30k.pkl')
svd = joblib.load('svd_model_30k.pkl')
latent_matrix_train = svd.transform(interaction_matrix.values)
latent_matrix_item = svd.components_
nn_model = joblib.load('nn_model_30k.pkl')

# ---------------------------------------------------------------------------------------------------------------------#
# ---------------------------------- Helper Functions ---------------------------------------------------------------#
# ---------------------------------------------------------------------------------------------------------------------#


# Function to recommend top-scored manga based on genres
def recommend_manga(genres, top_n=5):
    if not genres or not isinstance(genres, list):
        return "Please provide a list of genres."

    filtered_manga = manga_data[manga_data['Genres'].notna()]
    filtered_manga = filtered_manga[
        filtered_manga['Genres'].str.contains('|'.join(genres), case=False, na=False)
    ]

    top_manga = filtered_manga.sort_values(by='Score', ascending=False).head(top_n)
    return top_manga[['Title', 'Score', 'Genres', 'Link', 'Image Link']]

# Function to recommend manga with user input
def recommend_books_for_new_user(new_user_ratings, nn_model, interaction_matrix, svd, top_n=5):
    """
    Recommend manga based on a new user's ratings using collaborative filtering.
    """
    new_user_df = pd.DataFrame(new_user_ratings, index=['new_user'], columns=interaction_matrix.columns).fillna(0)
    new_user_vector = svd.transform(new_user_df.values)  # Shape: (1, n_components)
    distances, indices = nn_model.kneighbors(new_user_vector, n_neighbors=10)
    similar_users = interaction_matrix.index[indices.flatten()]

    weighted_sum = np.zeros(interaction_matrix.shape[1])
    similarity_sum = 0

    for idx, user in enumerate(similar_users):
        user_similarity_score = 1 - distances.flatten()[idx]  # Similarity = 1 - distance
        weighted_sum += interaction_matrix.loc[user].values * user_similarity_score
        similarity_sum += user_similarity_score

    if similarity_sum != 0:
        weighted_sum /= similarity_sum

    new_user_rated = new_user_df.loc['new_user']
    recommendations = pd.Series(weighted_sum, index=interaction_matrix.columns)
    recommendations = recommendations[new_user_rated == 0]

    return recommendations.sort_values(ascending=False).head(top_n).index.tolist()

# ---------------------------------------------------------------------------------------------------------------------#
# ---------------------------------------- Flask Routes -------------------------------------------------------------#
```

```python
# ------------------------------------------------------------------------------------------------------------------------#
@application.route('/')
def home():
    return render_template("home.html")

@application.route('/resume')
def resume():
    return render_template("resume.html")

@application.route('/projects')
def projects():
    return render_template("projects.html")

@application.route('/project_specific')
def project_specific():
    return render_template("project_specific.html")


@application.route('/manga_recommendation', methods=['GET', 'POST'])
def mangaRecommendation():
    recommendations = None
    manga_list = []
    genres = sorted(set(
        g.strip() for sublist in manga_data['Genres'].dropna().str.split(',') for g in sublist
    ))

    if request.method == 'POST':
        # Handle genre-based recommendations
        selected_genres = [
            request.form.get('genre_1'),
            request.form.get('genre_2'),
            request.form.get('genre_3')
        ]
        selected_genres = [g for g in selected_genres if g]

        if selected_genres:
            recommendations = recommend_manga(selected_genres, top_n=5)
            for _, row in recommendations.iterrows():
                # Process the image link for high resolution
                image_link = row['Image Link'] if pd.notna(row['Image Link']) else "https://via.placeholder.com/100x150"
                high_res_image_link = (
                    image_link.replace('/r/50x70', '') if pd.notna(image_link) and '/r/50x70' in image_link else image_link
                )
                manga_list.append({
                    "title": row['Title'],
                    "url": row['Link'],
                    "score": row['Score'],
                    "genres": row['Genres'],
                    "thumbnail": high_res_image_link
                })


        else:
            # Retrieve form data for manga ratings
            book_ratings = {}
            for i in range(1, 6):
                manga_title = request.form.get(f'manga_{i}')
                rating = request.form.get(f'rating_{i}')
                if manga_title and rating:
                    try:
                        book_ratings[manga_title.strip()] = int(rating)
                    except ValueError:
                        return "Invalid input. Please enter a number for ratings.", 400

            if book_ratings:
                # Generate recommendations using the collaborative filtering model
                recommended_titles = recommend_books_for_new_user(book_ratings, nn_model, interaction_matrix, svd, top_n=5)

                # Create list of manga with their titles, links, thumbnails, and scores
                for title in recommended_titles:
                    # Match on Title or English Title
                    manga_row = manga_image_links_df[
                        (manga_image_links_df['Title'].str.lower() == title.lower()) |
                        (manga_image_links_df['English Title'].str.lower() == title.lower())
                    ]

                    # Fallback values
                    link = "#"
                    high_res_image_link = "https://via.placeholder.com/100x150"
                    display_score = "No score available"

                    if not manga_row.empty:
```

```
                        # Use available data from the dataset
                        link = manga_row.iloc[0]['Link'] if pd.notna(manga_row.iloc[0]['Link']) else link
                        image_link = manga_row.iloc[0]['Image Link']
                        high_res_image_link = (
                            image_link.replace('/r/50x70', '') if pd.notna(image_link) else high_res_image_link
                        )
                        score = manga_row.iloc[0]['Score_y']
                        display_score = score if pd.notna(score) else display_score

                    # Add manga details to the list
                    manga_list.append({
                        "title": title,
                        "url": link,
                        "thumbnail": high_res_image_link,
                        "score": display_score
                    })

    return render_template("manga_recommendation.html", genres=genres, recommendations=manga_list)


# Run the app
if __name__ == '__main__':
    application.run(host="localhost", port=5000, debug=True)
    # application.run(debug=True)


```

~\Desktop\Portfolio-Project-Recommendation-System\final project folder\templates\home.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Manga Recommendations</title>
    <style>
        @import url('https://fonts.googleapis.com/css2?family=Roboto:wght@400&display=swap');

        body {
            font-family: 'Roboto', sans-serif;
            margin: 0;
            padding: 0;
            padding-bottom: 50px;
            display: flex;
            flex-direction: column;
            align-items: center;
            justify-content: flex-start;
            min-height: 100vh;
            background-color: #f1f1f1;
        }
        .light-blue-box {
            width: 100%;
            background-color: #dfdfdf;
            height: 400px;
            margin-top: 50px;
            position: absolute;
            top: 0;
            left: 0;
            z-index: -1;
        }
        .profile-container {
            text-align: center;
            padding: 0;
            margin-top: 20px;
        }
        .profile-img {
            width: 500px;
            height: 500px;
            border-radius: 50%;
            border: 5px solid #ccc;
            object-fit: cover;
            margin-top: 50px;
        }
        .profile-title {
            font-family: 'Lora';
            font-size: 3rem;
            font-weight: 700;
            margin-top: 20px;
            letter-spacing: 2px;
        }
        .profile-subtitle {
            font-size: 1.2rem;
            font-weight: 700;
            margin-top: 10px;
            color: #666;
            text-transform: uppercase;
        }
        .profile-description {
            max-width: 90%;
            margin: 20px auto;
            font-size: 1rem;
            line-height: 1.6;
            color: #444;
        }
        @media (min-width: 768px) {
            .profile-description {
                max-width: 1200px; /* For larger screens, limit the text width */
                font-size: 1.2rem; /* Increase font size for better readability */
            }
        }
        .link-container {
            display: flex;
            gap: 20px;
            justify-content: center;
            position: fixed;
            top: 0;
            width: 100%;
            background-color: #ffffff;
            padding: 10px 0;
```

```
            box-shadow: 0px 2px 5px rgba(0, 0, 0, 0.1);
            z-index: 1000;
        }
        .link-button {
            padding: 10px 20px;
            font-size: 1em;
            cursor: pointer;
            background-color: #667a91;
            color: #fff;
            border: none;
            border-radius: 5px;
            text-decoration: none;
            transition: background-color 1s, transform 1s;
        }
        .link-button:hover {
            background-color: #0057b30a;
            color: #ffffff09;
            transform: scale(1.05);
        }
    </style>
</head>
<body>

    <div class="link-container">
        <a href="/projects" class="link-button">Projects</a>
        <a href="/resume" class="link-button">Resume</a>
        <a href="/manga_recommendation" class="link-button">Manga Recommendation System</a>
    </div>

    <div class="light-blue-box"></div>
        <div class="profile-container">
            <img src="/static/profile_image.jpg" alt="Profile Photo" class="profile-img">
            <div class="profile-title"><span style="font-weight: 400;">Creative</span> <span style="font-weight: 700;">Portfolio</span></div>
            <div class="profile-subtitle">Data Scientist</div>
            <div class="profile-description">
                <h1>Hey, I’m Taylor Baker — Welcome to My Space!</h1>

                <h2>Academic Journey</h2>
                <p>I’m currently wrapping up my Master’s Degree in Data Science at Eastern University, set to graduate in December 2024. But my path here wasn’t exactly straight. My first degree was in fine art, where I learned to think creatively and approach problems from fresh perspectives. When COVID hit, I started taking online courses to explore something new, and that something turned out to be data science. Without anyone in my circle working in tech, I relied on self-study, stacking certifications in Data Science and Machine Learning/AI Engineering. Those early wins fueled my passion, and every late-night breakthrough reminded me I was on the right path.</p>

                <h2>Professional Interests</h2>
                <p>I’m passionate about machine learning, AI development, and making sure these systems are fair, inclusive, and accountable. For me, it’s not just about getting models to “work”, it’s about who they work for. I aim to build AI that benefits everyone, not just a select few. With hands-on experience in Python, SQL, and TensorFlow, I’m drawn to projects where I can solve challenging problems while ensuring the solutions are as ethical as they are effective. I’m especially interested in exploring how AI can be designed with transparency and accountability at its core.</p>

                <h2>The Human Behind the Code</h2>
                <p>At my core, I’m a curious problem-solver. Growing up, I was always fascinated by technology, but I didn’t have anyone around me working in tech, so I didn't know it was something I would enjoy. It wasn’t until I stumbled into coding during the pandemic that I realized this world was for me. That same drive to “figure things out” is what fuels my love for data science today. I’m constantly tinkering, learning, and asking, <em>“What if…?”</em> That curiosity shows up in my personal projects, like the user to user-based manga recommendation system I’m building using K-Nearest Neighbors. It’s a perfect mix of my technical skills and my love for storytelling.</p>

        </div>
    </div>


</body>
</html>

```

~\Desktop\Portfolio-Project-Recommendation-System\final project folder\templates\manga_recommendation.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Manga Recommendations</title>
    <style>
        @import url('https://fonts.googleapis.com/css2?family=Roboto:wght@400&display=swap');
        * {
        box-sizing: border-box; /* Includes padding, borders, and shadow in width/height */
    }

        body {
            font-family: 'Roboto', sans-serif;
            text-align: center;
            margin: 0 ;
            padding: 0;
            margin-top: 60px;
            padding-bottom: 100px;
        }
        .input-container {
            margin: 30px 0;
            text-align: left;
            margin-left: 10%;
            margin-right: 10%;
        }
        .input-group {
            margin-bottom: 15px;
        }
        label {
            font-weight: bold;
        }
        input[type="text"], input[type="number"], select {
            padding: 10px;
            font-size: 1em;
            width: calc(100% - 20px);
            max-width: 400px;
            margin-top: 5px;
        }
        button {
            padding: 10px 20px;
            font-size: 1em;
            cursor: pointer;
            background-color: #667a91;
            color: #fff;
            border: none;
            border-radius: 5px;
            text-decoration: none;
            transition: background-color 1s, transform 1s;
        }
        button:hover {
            background-color: #0057b30a;
            color: #ffffff09;
            transform: scale(1.05);
        }
        .recommendations {
            margin-top: 30px;
            text-align: center;
        }

        .recommendation-container {
            display: flex;
            flex-wrap: wrap;
            gap: 20px;
            justify-content: center;
        }

        .recommendation-box {
            display: block;
            border: 1px solid #ddd;
            border-radius: 5px;
            padding: 10px;
            width: 150px;
            text-align: center;
            text-decoration: none;
            color: inherit;
            transition: transform 0.3s, box-shadow 0.3s;
        }

        .recommendation-box:hover {
```

```
            transform: scale(1.05);
            box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.2);
        }

        .recommendation-box img {
            width: 100%;
            height: auto;
            margin-bottom: 10px;
            border: 1px solid #ddd;
            border-radius: 5px;
        }

        .recommendation-box h3 {
            font-size: 1em;
            color: #007BFF;
            font-weight: bold;
            margin: 5px 0;
        }

        .recommendation-box p {
            margin-top: 5px;
            font-size: 0.9em;
            color: #333;
        }
        .link-container {
            display: flex;
            justify-content: center;
            align-items: center;
            gap: 20px;
            position: fixed;
            top: 0;
            width: 100%;
            background-color: #ffffff;
            padding: 10px 0;
            box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.1);
            z-index: 1000;
            margin: 0;
        }
        .link-button {
            padding: 10px 20px;
            font-size: 1em;
            cursor: pointer;
            background-color: #667a91;
            color: #fff;
            border: none;
            border-radius: 5px;
            text-decoration: none;
            transition: background-color 1s, transform 1s;
        }
        .link-button:hover {
            background-color: #0057b30a;
            color: #ffffff09;
            transform: scale(1.05);
        }
    </style>
</head>
<body>
    <h1>Manga Recommendation System</h1>
    <div class="link-container">
        <a href="/projects" class="link-button">Projects</a>
        <a href="/resume" class="link-button">Resume</a>
        <a href="/" class="link-button">Home</a>
    </div>

    <div class="input-container">
        <form method="POST" action="/manga_recommendation">
            <!-- Genre Selection -->
            <h2>Select Genres for Recommendations</h2>
            <p style="margin-top: 30px; font-size: 1.2em; color: #555;">
                If you have not read any manga and are starting from scratch. Select from the drop down menu and
                pick a genre that you like. You can choose 1 to 3 genres and it will recommend 5 popular manga
                that contain the genres you selected.
            </p>
            <div class="input-group">
                <label for="genre_1">Genre 1:</label>
                <select id="genre_1" name="genre_1">
                    <option value="">-- Select Genre --</option>
                    {% for genre in genres %}
                    <option value="{{ genre }}">{{ genre }}</option>
                    {% endfor %}
                </select>
            </div>
```

```
            <div class="input-group">
                <label for="genre_2">Genre 2:</label>
                <select id="genre_2" name="genre_2">
                    <option value="">-- Select Genre --</option>
                    {% for genre in genres %}
                    <option value="{{ genre }}">{{ genre }}</option>
                    {% endfor %}
                </select>
            </div>
            <div class="input-group">
                <label for="genre_3">Genre 3:</label>
                <select id="genre_3" name="genre_3">
                    <option value="">-- Select Genre --</option>
                    {% for genre in genres %}
                    <option value="{{ genre }}">{{ genre }}</option>
                    {% endfor %}
                </select>
            </div>

            <!-- Ratings Input -->
            <h2>Rate Manga You Have Read</h2>
            <p style="margin-top: 30px; font-size: 1.2em; color: #555;">
                You can enter between 1 and 5 manga that you have read and you can rate each one between 0-10.
                This will give you a list of 5 manga recommendations based off 30,000 users data. If you don't
                like your recommendations or what to mix it up, change the ratings and you will get different results.
            </p>
            <div id="input-groups">
                <div class="input-group">
                    <label for="manga_1">Manga Title 1:</label>
                    <input type="text" id="manga_1" name="manga_1">
                    <label for="rating_1">Rating (0-10):</label>
                    <input type="number" id="rating_1" name="rating_1" min="0" max="10">
                </div>
                <div class="input-group">
                    <label for="manga_2">Manga Title 2:</label>
                    <input type="text" id="manga_2" name="manga_2">
                    <label for="rating_2">Rating (0-10):</label>
                    <input type="number" id="rating_2" name="rating_2" min="0" max="10">
                </div>
                <div class="input-group">
                    <label for="manga_3">Manga Title 3:</label>
                    <input type="text" id="manga_3" name="manga_3">
                    <label for="rating_3">Rating (0-10):</label>
                    <input type="number" id="rating_3" name="rating_3" min="0" max="10">
                </div>
                <div class="input-group">
                    <label for="manga_4">Manga Title 4:</label>
                    <input type="text" id="manga_4" name="manga_4">
                    <label for="rating_4">Rating (0-10):</label>
                    <input type="number" id="rating_4" name="rating_4" min="0" max="10">
                </div>
                <div class="input-group">
                    <label for="manga_5">Manga Title 5:</label>
                    <input type="text" id="manga_5" name="manga_5">
                    <label for="rating_5">Rating (0-10):</label>
                    <input type="number" id="rating_5" name="rating_5" min="0" max="10">
                </div>
            </div>
            <button type="submit">Get Recommendations</button>
        </form>
    </div>

    {% if recommendations %}
    <div class="recommendations">
        <h2>Recommended Manga for You:</h2>
        <div class="recommendation-container">
            {% for manga in recommendations %}
            <a href="{{ manga.url }}" target="_blank" class="recommendation-box">
                <img src="{{ manga.thumbnail }}" alt="{{ manga.title }}">
                <h3>{{ manga.title }}</h3>
                <p>Score: {{ manga.score }}</p>
            </a>
            {% endfor %}
        </div>
    </div>
    {% endif %}
</body>
</html>
```

~\Desktop\Portfolio-Project-Recommendation-System\final project folder\templates\resume.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Resume - Taylor Baker</title>
    <style>
        body {
            font-family: 'Roboto', sans-serif;
            max-width: 100%;
            margin: 0 auto;
            text-align: center;
        }
        .pdf-container {
            max-width: 100%;
            overflow: hidden;
        }
        .link-container {
            display: flex;
            gap: 20px;
            justify-content: center;
            position: fixed;
            top: 0;
            width: 100%;
            background-color: #ffffff;
            padding: 10px 0;
            box-shadow: 0px 2px 5px rgba(0, 0, 0, 0.1);
            z-index: 1000;
        }
        .link-button {
            padding: 10px 20px;
            font-size: 1em;
            cursor: pointer;
            background-color: #667a91;
            color: #fff;
            border: none;
            border-radius: 5px;
            text-decoration: none;
            transition: background-color 1s, transform 1s;
        }
        .link-button:hover {
            background-color: #0057b30a;
            color: #ffffff09;
            transform: scale(1.05);
        }
    </style>
</head>
<body>
    <h1>My Resume</h1>
    <div class="link-container">
        <a href="/projects" class="link-button">Projects</a>
        <a href="/resume" class="link-button">Resume</a>
        <a href="/" class="link-button">Home</a>
    </div>
    <!-- Displaying the PDF directly in the webpage -->
    <div class="pdf-container">
        <embed src="/static/Taylor Baker – Data Science- resume 2024.pdf" type="application/pdf" width="100%" height="800px">
    </div>

    <div style="margin-top: 30px;">
        <a href="/" style="display: inline-block; padding: 12px 25px; font-size: 1em; background-color: #5eccd5; color: #fff; text-decoration: none; border-radius: 5px; transition: background-color 0.3s, transform 0.3s;" onmouseover="this.style.backgroundColor='#ffffff'; this.style.color='#5eccd5';" onmouseout="this.style.backgroundColor='#5eccd5'; this.style.color='#ffffff';">
            Back to Home
        </a>
    </div>
</body>
</html>
```

~\Desktop\Portfolio-Project-Recommendation-System\final project folder\templates\projects.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Projects - Manga Recommendation System</title>
    <style>
        @import url('https://fonts.googleapis.com/css2?family=Roboto:wght@400&display=swap');

        body {
            font-family: 'Roboto', sans-serif;
            margin: 0;
            padding: 0;
            display: flex;
            flex-direction: column;
            align-items: center;
            justify-content: flex-start;
            min-height: 100vh;
            background-color: #fafafa; /* Off white background */
        }
        .container-wrapper {
            width: 100%;
            background-color: #f6ebfd;
            padding: 30px 0;
            box-sizing: border-box;
        }
        .container {
            display: flex;
            flex-direction: row;
            align-items: center;
            justify-content: center;
            gap: 40px;
            margin-top: 50px;
            max-width: 1200px;
            padding: 20px;
        }
        .project-image {
            width: 300px;
            height: 300px;
            border-radius: 10px;
            object-fit: cover;
            cursor: pointer;
            transition: transform 0.3s;
        }
        .project-image:hover {
            transform: scale(1.05);
        }
        .project-description {
            max-width: 600px;
            font-size: 1.2rem;
            line-height: 1.6;
            color: #444;
        }
        .link-container {
            display: flex;
            gap: 20px;
            justify-content: center;
            position: fixed;
            top: 0;
            width: 100%;
            background-color: #ffffff;
            padding: 10px 0;
            box-shadow: 0px 2px 5px rgba(0, 0, 0, 0.1);
            z-index: 1000;
        }
        .link-button {
            padding: 10px 20px;
            font-size: 1em;
            cursor: pointer;
            background-color: #667a91;
            color: #fff;
            border: none;
            border-radius: 5px;
            text-decoration: none;
            transition: background-color 1s, transform 1s;
        }
        .link-button:hover {
            background-color: #0057b30a;
            color: #ffffff09;
            transform: scale(1.05);
```

```
        }
    </style>
</head>
<body>
    <h1>My Projects</h1>

    <div class="link-container">
        <a href="/" class="link-button">Back to Home</a>
        <a href="/resume" class="link-button">Resume</a>
        <a href="/projects" class="link-button">Projects</a>
    </div>
    <div class="container-wrapper">
        <div class="container">
            <a href="/project_specific">
                <img src="/static/manga_pic.jpg" alt="Manga Recommendation System" class="project-image">
            </a>
            <div class="project-description">
                <h2><a href="/manga_recommendation">Manga Recommendation System</a></h2>
                <p>A collaborative filtering-based recommendation engine for manga. This system uses user preferences to suggest new manga titles that match the tastes of similar users.</p>
            </div>
        </div>
    </div>

</body>
</html>


```