Analysis of data from Goodreads

Shambhavi Agrawal
B.Tech 22
Indian Institute of Technology
Gandhinagar

***Abstract*—This paper analyses various datasets obtained from Goodreads on books. It answers scientific questions and proves/disproves proposed hypotheses using python codes, plots, graphs etc. on these datasets. There are a few unanswerable questions included.**

I. Overview of the Dataset

The dataset includes specific data about books, ratings, user, author etc. One of the dataframes has information about the book's original title, author and publication year. These are all linked with the help of book IDs, goodreads book ID, work ID and best book ID. It includes the book counts which gives information about the number of editions a particular book is available in. These can be different editions in the same language or translations of the book in different languages. It also tells the language of the book written. Most important, it has the average ratings of each book, the ratings count, work ratings count. The next dataframe gives links the user ID with the respective book ID for the book which the user has rated, along with ratings. Another dataframe includes book IDs of books which users have marked as 'to read'. The dataset has includes tags provided by the user for various books which gives information about the genre of the book, such as fictional, romance, comedy, contemporary, cookbooks, classics, crime, fantasy, horror, mystery etc. A lot of information can be extracted from this dataset and useful analysis can be done.

II. Scientific Questions And Hypotheses

A. What is the trend of average rating of books? Selecting a Template (Heading 2)

Users rate books on a variety of aspects. Some books get better ratings than others. We would like to know if there is any pattern between the number of users giving a specific rating for books. In other words, are some ratings given more than others? Could there be a pattern in the curve between the average rating of books and the number of users giving that particular rating? Does the Probability Density Function look like any of the standard curves? What is the maximum average rating calculated from users' ratings in this dataset?

B. Are the number of books being published increasing over the years?

A number of books get published every year. We would like to explore the trend of the number of books being published from the past years starting from 20th century (since the number of books published before that is very less as compared to the past few centuries). Since the literacy rate is increasing and the number of people reading books is increasing, is there an increase in the number of books getting published year by year? What inference can be drawn from this data with regards to mass production?

C. Are authors who write more books, better? Authors' books' ratings dependance on the number of books written

Some authors write just a few books while others write as many as more than 50. Naturally, one would think that the more books an author publishes the more popular/good writer he/she must be. Is there a pattern between the number of books published by an author and the mean value of the average ratings of the books published by him/her? According to our assumption as the number of books published increases for a particular author, the mean ratings of the books published by him/her should also increase.

D. Do users rating frequently rate different than others?

Many users are frequent raters while others are not so frequent. If a user rates frequently that means that he/she is an avid reader, which in turn means that he/she will have a better idea about the quality of books and its content and will rate more critically. Does the graph between the frequency of raters and the mean of the ratings they have given, show any pattern? More specifically, can it prove the above hypotheses?

E. Does more editions of a book mean more popularity/better ratings?

When a book is published, it gets published by different publishers, it gets revised, it gets translated into different languages and hence many editions are released. A book having more number of editions should mean that is more popular amongst the public and hence should have higher average ratings. Can the graph between the number of editions of a particular book and the average rating received by it, show any such variation?

F. What are the top books in the market?

Suppose you want to start reading books but don't know where to start. You usually ask friends, book store owners or the internet to find out the most interesting and highest rated books. Can we find out from this dataset, what are the top 10 books receiving highest ratings? Do many of these books belong to a particular series or are there individual books? Can we conclude that the series is popular?

G. How many books are there in a particular series?

Suppose you want to binge read a series, you would want to know the number of books that series contains and plan accordingly. Some series contain more than 20 books while others have as low as 3. Can we find out the number of books in a series from this dataset?

III. Libraries and Functions

To analyse the data quite a few libraries need to be installed and imported before writing the python code. A few of the libraries that would be helpful are, Matplotlib, Pandas, NumPy, Regular Expressions. Matplotlib is used to plot graphs, curves of the data. It has many features such as labelling the axes, giving title, colour, size. Various graphs like bar graph, line graph, scatter, histogram etc. can be visualised with the help of this library. Pandas is an extremely helpful tool for data analysis. It has many functions that can effectively help organise and view the data from different perspectives. Functions like DataFrame and Series help organise data. Data can be extracted using loc, head, tail functions. It can be sorted and changed using queries and other functions like value counts etc. The mean value (mean), standard deviation (std), variance (var), maximum (max), minimum (min) of a dataset can be easily calculated using the respective functions written inside the parenthesis. It can read csv files and store the data in DataFrames. It can plot various types of graphs including Probability Density Functions (kde) NumPy helps create arrays, matrices and perform basic mathematical operations. Regular expressions helps work on the strings i.e. title of books etc. A few other libraries that would be useful in analysis would be scipy for optimizing data, scikit-learn for advanced data analysis etc.

.

IV. Analysing Data and Answering Questions/Hypotheses

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

A. Analysing Trend of Average Rating of Books

1) Procedure

We first plot the histogram of the average rating of books vs the frequency of these ratings i.e. the number of times users have rated that particular rating. Next we can compute the maximum average rating and the mean average rating using pandas functions. Also, we plot the Probability Density Function of the average ratings using kde function for better evaluation of data. The following code snippet can be used:

```
books = pd.read_csv("books.csv")
max = books.average_rating.max() #Highe
st average rating.
print(f"The highest average rating is:
{max}")
mean =  books.average_rating.mean() # M
ean value
print(f"The mean rating value is: {mean
}")
ax = books.average_rating.plot(kind = "
hist", bins = 50, xlabel = "Average Rat
ing", ylabel = "Frequency", title = "Fr
equency of average rating of books")
ax.set(xlabel = "Average Rating")
graph = books.average_rating.plot.kde()
```

2) Observation and Results

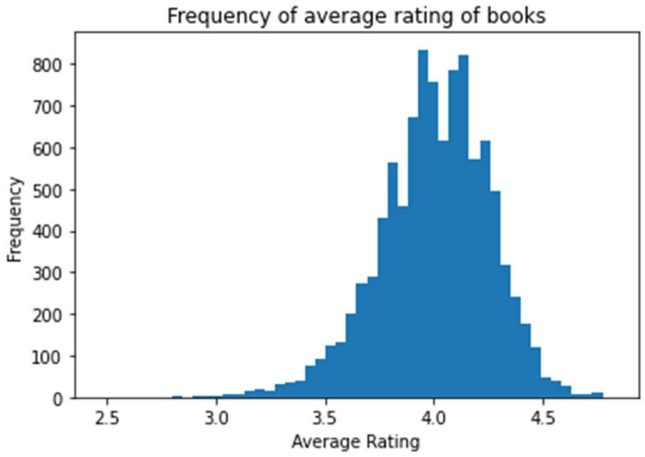

Fig. 1. Histogram of frequency of average rating of books

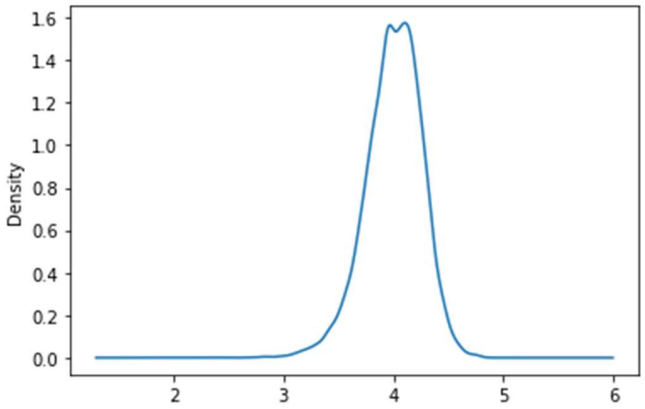

Fig. 2.: Probability Density Function of average rating

From the code, we get the highest average rating given by users to be computed as 4.82, and the mean value as 4.002191000000001.

We can observe and verify the above data from Fig. 1. as well. From the graph we can see that average rating ranging from 3.8 to 4.2 are the most frequent ratings given

by users. The most interesting and important point to note from Fig. 2. is that the PDF of the average rating looks similar (not same) to a Gaussian curve i.e. it seems like the datapoints of average rating of books is similar to a Normal Distribution. Also, the peak of the graph is very near to the mean value calculated, and we know that a Gaussian Curve is symmetric about the mean. The frequency of average ratings is maximum near the mean value and is decreases as we go farther away from the mean. Thus we can say that most people rate books approximately as 4 and near to that.

B. *Analysis of the number of books published over the past years.*

1) Procedure

We first extract publication years in the 20th and 21st century from the dataset by creating a query. Next, we sort and compute from this data, the number of books published in all those years. (This is done by counting the frequency of the appearance of a particular year in the dataset). This data is then plotted as a bar graph showing year vs the number of books published in that year. Next we compute the year in which the maximum number of books were published and the count of the books as well. The following code snippet can be used:

```
publish = books.query("`original_public
ation_year` > 1900").groupby('original_
publication_year').size().plot(kind = "
bar", title = "Number of books publishe
 in the 20th and 21st century", figsize
 = (20, 5), rot = 90)
publish.set(xlabel = "Publication Year"
, ylabel = "Number of books")
print(books.query("`original_publicatio
n_year` > 1900").groupby('original_publ
ication_year').size(), end = "\n\n")
maxyear = books.query("`original_public
ation_year` > 1900").groupby('original_
publication_year').size().idxmax() #yea
r in which max books were published.
maxbooks = books.query("`original_publi
cation_year` > 1900").groupby('original
_publication_year').size().max() #max b
ooks published in year.
top = books.original_publication_year.v
alue_counts().head() # top 5 years in w
hich maximum books were published
minyear = books.query("`original_public
ation_year` > 1900").groupby('original_
publication_year').size().idxmin() #yea
r in which min books were published.
minbooks = books.query("`original_publi
cation_year` > 1900").groupby('original
_publication_year').size().min() # min
books published
print(f"The maximum number of books wer
e published in the year {maxyear} and t
he count is {maxbooks}")
print(f"The minimumm number of books we
re published in the year {minyear} and
the count is {minbooks}")
print(f"The top five years in publishin
g maximum books are:\n{top}")
```

2) Observation and Results

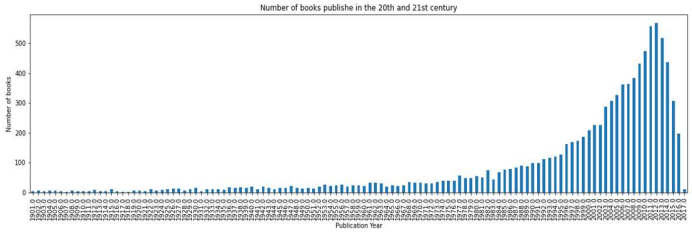

Fig. 3. Bar graph showing number of books published in the 20th and 21st century.

From the code the maximum number of books is computed to be published in the year 2012, and the count of the books is 568. Similarly, the minimum number of books was published in the year 1907, and the count of the books is 2. The top five years in publishing maximum number of books are: 2012, 2011, 2013, 2010, 2014 in decreasing order.

From the graph it can be observed that the number of books published increases over the years till the year 2012, after that there is a decrease till the year 2017. This proves our hypothesis that the publication of books increased over the years but to some extent for which the decrease has to be accounted for. An interesting although not very surprising fact is that the increase in the publication of books is similar to an exponential increase i.e. there has been a rapid increase in publication. This increase might be explained by reasons stated in our hypothesis that the literacy rates increase, number of people reading books increase and the authors writing books also increase over the years. Though the decrease in the publication of books after the year 2012 is something unexpected.

3) For papers with more than six authors: Add author names horizontally, moving to a third row if needed for more than 8 authors.

4) For papers with less than six authors: To change the default, adjust the template as follows.

a) Selection: Highlight all author and affiliation lines.

b) Change number of columns: Select the Columns icon from the MS Word Standard toolbar and then select the correct number of columns from the selection palette.

c) Deletion: Delete the author and affiliation lines for the extra authors.

C. *Analysing the realtion between the number of books an author writes and the ratings his/her books receive*

1) *Procedure*

First, we compute the number of books written by different authors using value counts function. Next, since we want to analyze data for authors who have written multiple books, we specify that the count of the books should be greater than one. We then plot a bar graph of the authors vs the number of books written by them. Then using a for loop we calculate the mean of the average ratings received for all the books written by a particular author. Similarly, we calculate the standard deviation. The following code snippet can be used:

```
smpbooks = books.loc[1:1000]
unique = smpbooks.authors.value_counts(
)
unique = unique[unique.gt(1)]
print(f"Number of books written by auth
ors:\n{unique}", end = "\n\n")
gph = unique.plot(kind = "bar", figsize
 = (20, 5), rot = 90, title = "Number o
f books written by authors")
gph.set(xlabel = "Author", ylabel = "Nu
mber of books")
idc = unique.index
mnbooks = pd.Series(np.zeros(len(idc)),
 index = idc)
stdbooks = pd.Series(np.zeros(len(idc))
, index = idc)
for name in idc:
  mnbooks[name] = smpbooks[smpbooks['au
thors'] == name]["average_rating"].mean
()
print(f"Mean of all the average ratings
 recieved for all the books written by
the author:\n{mnbooks}", end = "\n\n")
for name in idc:
  stdbooks[name] = smpbooks[smpbooks['a
uthors'] == name]["average_rating"].std
()
print(f"Standard deviation of all the a
verge ratings recieved for all the book
s written by the author:\n{stdbooks}")
ax = mnbooks.plot(kind = "bar", figsize
 = (20, 5), rot = 90, title = "Mean rat
ings of all books written by author")
ax.set(xlabel = "Author", ylabel = "Mea
n rating")
ay = stdbooks.plot(kind = "bar", figsiz
e = (20, 5), rot = 90, title = "Standar
d deviation of ratings of all books wri
tten by author")
ay.set(xlabel = "Author", ylabel = "Sta
ndard deviation of ratings")
plt.plot(unique, mnbooks, ',')
plt.xlabel("Number of books")
plt.ylabel("Mean ratings of authors")
plt.title("Mean ratings of authors vs n
umber of books")
```

2) *Observation and Results*

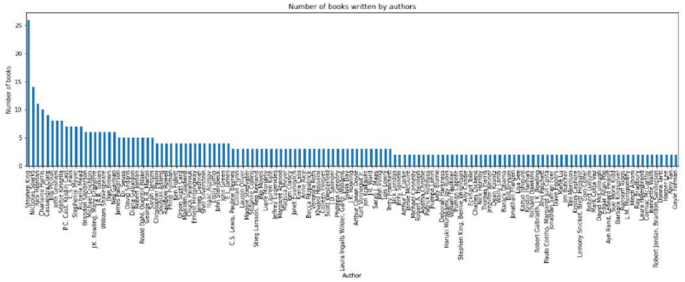

Fig. 4. Bar graph showing number of books written by authors

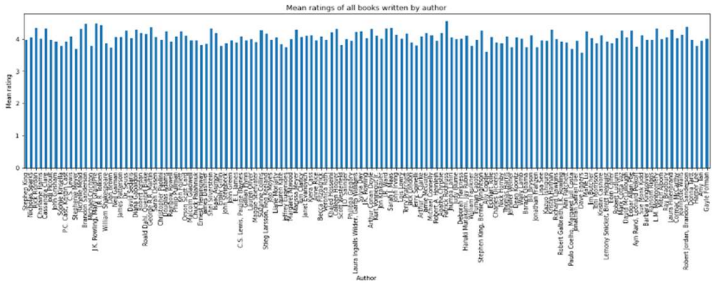

Fig. 5. Graph showing mean of average ratings received for all the books written by the author.

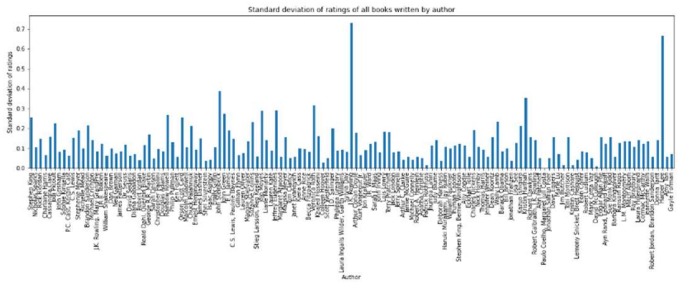

Fig. 6. Graph showing standard deviation of average ratings received for all the books written by the author.

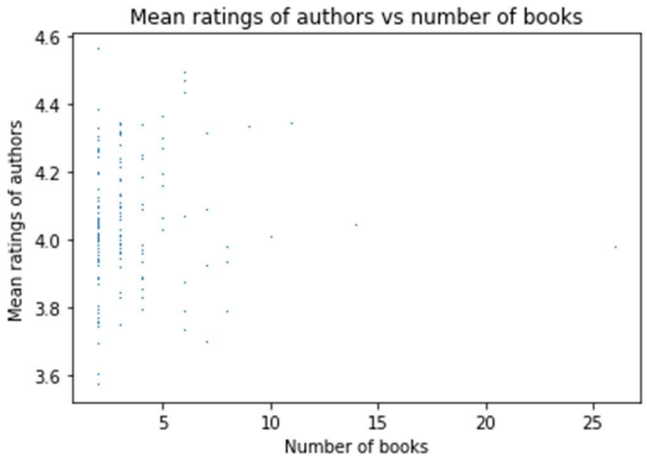

Fig. 7. Graph showing variation between the number of books published and the mean rating the corresponding author receives.

The graph in Fig. 4. shows the variation in the number of books written by different authors. A few authors have written as many as 26 books while some have written as few as only 2 books. Here we can see that there seems to be no significant relation between the number of books published and the mean value of the average ratings of all the books written by an author. Thus, our hypothesis is proved to be wrong. We expected better ratings for authors who have written more number of books. Clearly from Fig. 7. We can see that this is not the case. Even the graph for standard deviation i.e. Fig. 6. does not provide a good analysis of the situation We probably need more datapoints specifically for authors who have written many books, to get a good statistical inference.

Headngs, or heads, are organizational devices that guide the reader through your paper. There are two types: component heads and text heads.

Component heads identify the different components of your paper and are not topically subordinate to each other. Examples include Acknowledgments and References and, for these, the correct style to use is “Heading 5”. Use “figure caption” for your Figure captions, and “table head” for your table title. Run-in heads, such as “Abstract”, will require you to apply a style (in this case, italic) in addition to the style provided by the drop down menu to differentiate the head from the text.

Text heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic. If there are two or more sub-topics, the next level head (uppercase Roman numerals) should be used and, conversely, if there are not at least two sub-topics, then no subheads should be introduced. Styles named “Heading 1”, “Heading 2”, “Heading 3”, and “Heading 4” are prescribed.

D. Analysing the rating styles of frequent raters.

1) Procedure

To avoid too long computational time, we have a taken a subset of the dataset consisting of ratings given by users. Next, we compute the number of ratings given by each user. Then using mean function, we calculate the mean of the ratings given by a particular user. Similarly, we calculate the standard deviation and the variance of the datapoints. Then we plot the graphs of these calculated data. The following code snippet can be used:

```
ratings = pd.read_csv("ratings.csv")
smpratings = ratings.loc[1:100000]
users = smpratings.user_id.value_counts
()
idc = users.index
mnratings = pd.Series(np.zeros(len(idc)
), index = idc)
for user in idc:
  mnratings[user] = smpratings[smpratin
gs['user_id'] == user]["rating"].mean()
stdratings = pd.Series(np.zeros(len(idc
)), index = idc)
for user in idc:
  stdratings[user] = smpratings[smprati
ngs['user_id'] == user]["rating"].std()
varratings = pd.Series(np.zeros(len(idc
)), index = idc)
for user in idc:
  varratings[user] = smpratings[smprati
ngs['user_id'] == user]["rating"].var()
plt.plot(users, mnratings, ",")
plt.xlabel("Number of ratings by
users")
plt.ylabel("Mean value of average ratin
gs by user")
plt.title("Frequency of users and the m
ean value of ratings by them")
plt.plot(users, stdratings, ",")
```

```
plt.xlabel("Number of ratings by
users")
plt.ylabel("Standard deviation of avera
ge ratings by user")
plt.title("Frequency of users and the s
tandard deviation of ratings by them")
plt.plot(users, varratings, ",")
plt.xlabel("Number of ratings by
users")
plt.ylabel("Variance of average ratings
 by user")
plt.title("Frequency of users and the v
ariance of ratings by them")
```

2) Observations and Results

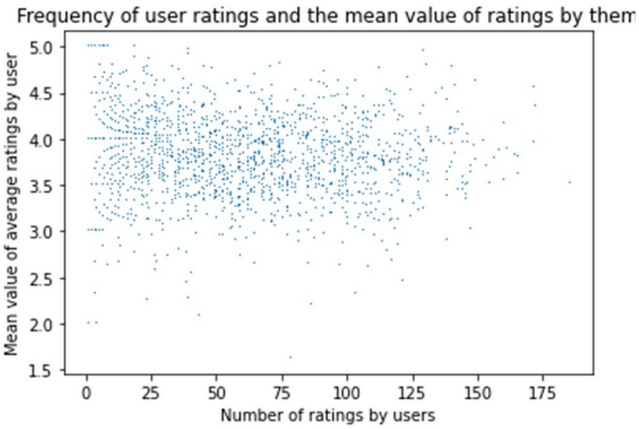

Fig. 8. Graph of the number of ratings by users and the mean value of the ratings by them.

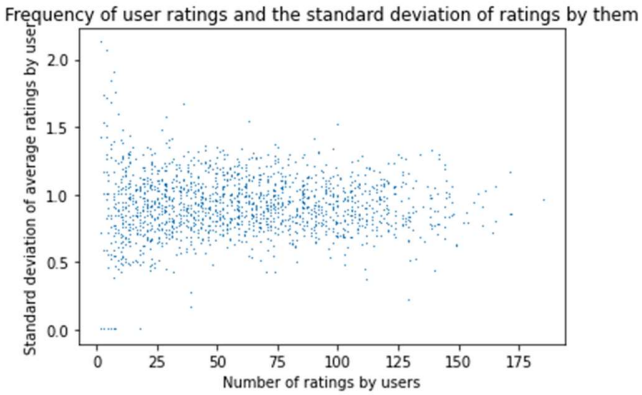

Fig. 9. Graph of the number of ratings by users and the standard deviation of the ratings by them.

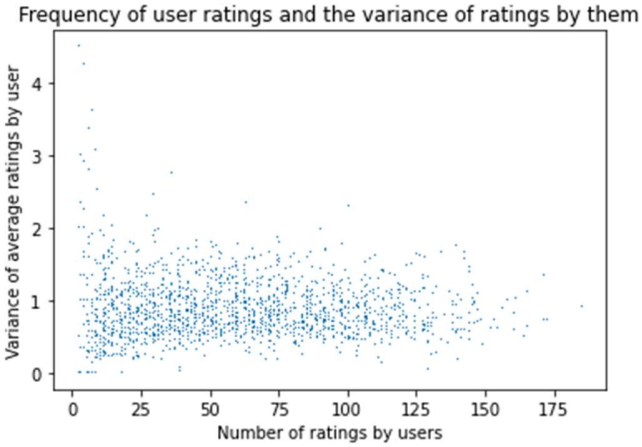

Fig. 10. Graph of the number of ratings by users and the variance of the ratings by them.

In the above graphs we can see that there is not much of a relation between the number of ratings by users and the mean of all the ratings by them. Thus, our hypothesis stating that users who rate more frequently, will rate more critically is proved wrong in this case. In Fig. 8., we can see that even very frequent raters have mean ratings in the range of 3 to 4.5. Although, we can see a slight reduction in the mean value when the number of ratings increase, i.e. the less users have their mean ratings close to 5. Also, we can see short straight lines at 3, 4 and 5 in Fig. 8. When the number of ratings is less by use, this is expected since if the number of ratings is very less, there will not be much variation in the mean value.

E. Relation between the number of editions of a book and the popularity/rating of it.

1) Procedure

Here a graph is plotted using matplotlib showing the relation between the number of editions of a book and the average rating received by it from various users. The following code snippet can be used:

```
books = pd.read_csv("books.csv")
plt.plot(books.books_count, books.avera
ge_rating, ',')
plt.xlabel("Number of editions of a boo
k")
plt.ylabel("Averge rating recieved by t
he book")
plt.title("Number of editions vs the av
erage rating recieved by book")
```

2) Observations and Results

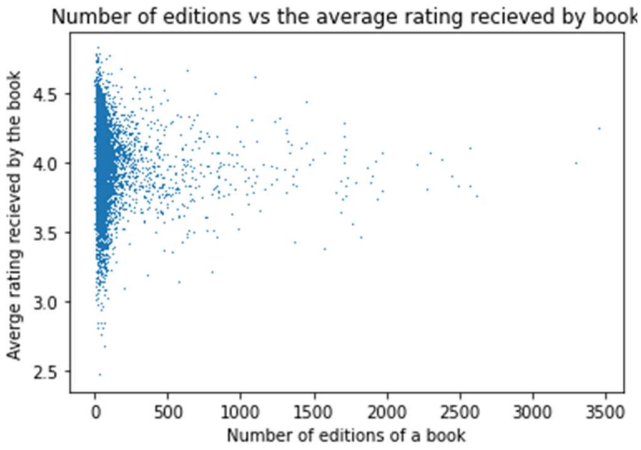

Fig. 11. Graph between the number of editions of books published and the average rating received by the book.

Here, it is difficult to draw statistical inference as there are less datapoints. From the graph we can observe that for less number of editions of books the average rating varies from 3 to 4.5, whereas for books having more editions, the average rating is close to 4. We can infer that books having more editions could be considered to have more stable and good ratings whereas for books having lesser editions, it really does depend on the quality of the book. Hence, our hypothesis can be considered true to some extent.

F. *Top books in the market*

1) *Procedure*

Select the top 10 highest rated books using pandas function nlargest, The following code snippet can be used:

```
toprate = books.nlargest(13, "average_rating").dropna()
toprate = toprate[["books_count", "authors", "original_publication_year", "original_title", "average_rating"]]
toprate
```

2) *Observations and Results*

	books_count	authors	original_publication_year	original_title	average_rating
3627	14	Bill Watterson	2005.0	The Complete Calvin and Hobbes	4.8
861	34	Brandon Sanderson	2014.0	Words of Radiance	4.7
8853	6	Francine Rivers	1993.0	Mark of the Lion Trilogy	4.7
4482	21	Bill Watterson	1996.0	It's a Magical World: A Calvin and Hobbes Coll...	4.7
421	76	J.K. Rowling	1998.0	Complete Harry Potter Boxed Set	4.7
6360	22	Bill Watterson	1996.0	There's Treasure Everywhere: A Calvin and Hobb...	4.7
3752	6	J.K. Rowling	2005.0	Harry Potter Collection (Harry Potter, #1-6)	4.7
6589	21	Bill Watterson	1990.0	The Authoritative Calvin and Hobbes	4.7
6919	19	Bill Watterson	1992.0	The Indispensable Calvin and Hobbes: A Calvin ...	4.7
9565	24	Bill Watterson	1992.0	Attack of the Deranged Mutant Killer Monster S...	4.7

Here we can see that top ten books in the table. From the table we can infer that Calvin and Hobbes by Bill Watterson and Harry Potter by J.k.Rowling seem to be the most popular.

G. *Number of books in a series*

1) *Procedure*

Use regular expressions, slicing and split to get the name of the series from the title. Then count the number of repititions of the name of the series by creating a dataset. The following code snippet can be used:

```
import re
l = []
for title in books.title:
  if len(title := title.split("(")) == 2:
    title = title[1]
  else: continue
  for t in title.split("; "):
    if matches := re.search(r"(.+), *#[0-9]+", t):
      l.append(matches.groups()[0])

series = pd.DataFrame(l).value_counts()
print(series)
```

2) *Observations and Results*

Since the dataframe printed out is very big, a short sample of the final data extracted is given below:

```
In Death 41
Discworld 40
Stephanie Plum 28
Harry Bosch Universe 26
Hercule Poirot 25
```

One can extract useful information from this dataset. This gives the number of books published in a series.

V. Unanswered Questions

1) If the number of volumes in a series is higher is the average rating for that series high as well.

2) Is the sequel of the books better than its prequel?

3) If there are multiple authors, does it affect the average rating of the books in any way?

4) Does the length of the title affect the average rating?

5) If a book has a subtitle, does it affect the average rating in any way?

6) Categorising books based on their book tags in different genres and comparing popularity.

ACKNOWLEDGMENT

The author would like to thank Prof. Shanmuganathan Raman and his entire team for their support and guidance in this project.

REFERENCES

[1] "User Guide — Pandas 1.5.3 Documentation," n.d. https://pandas.pydata.org/docs/user_guide/index.html#user-guide.

[2] "Matplotlib Documentation — Matplotlib 3.7.0 Documentation," n.d. https://matplotlib.org/stable/index.html.