# Comprehensive Analysis of Book Ratings and Trends

In [1]:
```
# Edit all the Mardown cells below with the appropriate information
# Run all cells, containing your code
# Save this Jupyter with the outputs of your executed cells
# PS: Save again the notebook with this outcome.
# PSPS: Don't forget to include the dataset in your submission
```

**Team:**

- Ahmed Husain

**Course:** CISD 43 – BIG DATA (Spring, 2024)

## Problem Statement

*This project is about analyzing a dataset of books to understand various attributes and their relationships. I will be using various methodologies to explore and visualize the data.

## Keywords:

```
Book analysis, Goodreads, data exploration
```

## Required packages

- Additional packages required are: bash, pip install pandas numpy matplotlib seaborn scikit-learn

In [9]:
```
import pandas as pd
import pymongo
from pymongo import MongoClient
```

## Methodology

1. Introduction

   In this project, I will analyze a dataset of books to uncover key insights and trends within the data. The analysis will follow a structured approach, beginning with loading the dataset into a manageable format for examination. Next, I will explore the data to understand its structure and the relationships between different attributes, such as publication year, authors, ratings, and more. Following this, I will visualize key attributes and trends to provide a clearer picture of the underlying patterns and distributions. After gaining an initial understanding of the data, I will build predictive models to estimate book rating patterns based on various features. Lastly, I will introduce and explain the topics and methodologies

used throughout the project, including data cleaning, exploratory data analysis, data visualization, and predictive modeling, to provide a comprehensive overview of the process and findings.

In this project, I will be using the steps below:
a.Load the dataset.
b.Explore the data to understand the structure and relationships.
c.Visualize key attributes and trends.
d.Build predictive models to understand rating patterns.

2. Exploratory steps involved in this project
   a.Data Cleaning
   b.Exploratory Data Analysis (EDA)
   c.Data Visualization
   d.Predictive Modeling

   - Model 1
     - KNN For the first predictive model, I implemented the K-Nearest Neighbors (KNN) algorithm to estimate the average rating of books based on features such as ratings count, work ratings count, and text reviews count. The KNN algorithm predicts a book's rating by averaging the ratings of its five nearest neighbors, determined by the similarity of their features. I set the number of neighbors (k) to 5, which provided a balance between bias and variance. The KNN model achieved a Mean Squared Error (MSE) of 0.0696, indicating a reasonable level of accuracy in predicting book ratings, although it was outperformed by the Linear Regression model in terms of predictive performance.
   - Model 2
     - Linear Regression For the second predictive model, I implemented Linear Regression to estimate the average rating of books using features such as ratings count, work ratings count, and text reviews count. Linear Regression works by fitting a linear equation to the observed data, which allows it to model the relationship between the input features and the target variable. This model achieved a Mean Squared Error (MSE) of 0.0596, which is lower than that of the KNN model, indicating a better fit to the data. The results suggest that the relationship between the input features and average book ratings is approximately linear, making Linear Regression a suitable choice for this dataset.

## Data Loading and Exploration

In [10]:
```
# Replace the following with your MongoDB connection string
MONGO_URI = "mongodb://localhost:27017/"

# Connect to MongoDB server
client = MongoClient(MONGO_URI)
```

```
# Access the 'Books' database
db = client['Books']

# Access the 'Books' collection
books_collection = db['Books']

# Load the dataset from MongoDB
data = list(books_collection.find())
df = pd.DataFrame(data)

# List all field names (column names)
field_names = df.columns.tolist()
print("Field Names:", field_names)

# Display the top 50 rows of the DataFrame
top_50_rows = df.head(50)
top_50_rows


# Read the CSV file into a DataFrame
df = pd.read_csv(file_path)

# List all field names (column names)
field_names = df.columns.tolist()
print("Field Names:", field_names)

# Display the top 50 rows of the DataFrame
top_50_rows = df.head(50)
top_50_rows
```

```
Field Names: ['_id', 'book_id', 'goodreads_book_id', 'best_book_id', 'work_id', 'book
s_count', 'isbn', 'isbn13', 'authors', 'original_publication_year', 'original_title',
'title', 'language_code', 'average_rating', 'ratings_count', 'work_ratings_count', 'w
ork_text_reviews_count', 'ratings_1', 'ratings_2', 'ratings_3', 'ratings_4', 'ratings
_5', 'image_url', 'small_image_url']
Field Names: ['book_id', 'goodreads_book_id', 'best_book_id', 'work_id', 'books_coun
t', 'isbn', 'isbn13', 'authors', 'original_publication_year', 'original_title', 'titl
e', 'language_code', 'average_rating', 'ratings_count', 'work_ratings_count', 'work_t
ext_reviews_count', 'ratings_1', 'ratings_2', 'ratings_3', 'ratings_4', 'ratings_5',
'image_url', 'small_image_url']
```

Out[10]:

| | book_id | goodreads_book_id | best_book_id | work_id | books_count | isbn | isbn13 | |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 2767052 | 2767052 | 2792775 | 272 | 439023483 | 9.780439e+12 | |
| **1** | 2 | 3 | 3 | 4640799 | 491 | 439554934 | 9.780440e+12 | J. |
| **2** | 3 | 41865 | 41865 | 3212258 | 226 | 316015849 | 9.780316e+12 | |
| **3** | 4 | 2657 | 2657 | 3275794 | 487 | 61120081 | 9.780061e+12 | |
| **4** | 5 | 4671 | 4671 | 245494 | 1356 | 743273567 | 9.780743e+12 | |
| **5** | 6 | 11870085 | 11870085 | 16827462 | 226 | 525478817 | 9.780525e+12 | J |
| **6** | 7 | 5907 | 5907 | 1540236 | 969 | 618260307 | 9.780618e+12 | J.R |
| **7** | 8 | 5107 | 5107 | 3036731 | 360 | 316769177 | 9.780317e+12 | J. |
| **8** | 9 | 960 | 960 | 3338963 | 311 | 1416524797 | 9.781417e+12 | |
| **9** | 10 | 1885 | 1885 | 3060926 | 3455 | 679783261 | 9.780680e+12 | Ja |
| **10** | 11 | 77203 | 77203 | 3295919 | 283 | 1594480001 | 9.781594e+12 | |
| **11** | 12 | 13335037 | 13335037 | 13155899 | 210 | 62024035 | 9.780062e+12 | |
| **12** | 13 | 5470 | 5470 | 153313 | 995 | 451524934 | 9.780452e+12 | O |
| **13** | 14 | 7613 | 7613 | 2207778 | 896 | 452284244 | 9.780452e+12 | |
| **14** | 15 | 48855 | 48855 | 3532896 | 710 | 553296981 | 9.780553e+12 | A |
| **15** | 16 | 2429135 | 2429135 | 1708725 | 274 | 307269752 | 9.780307e+12 | La |
| **16** | 17 | 6148028 | 6148028 | 6171458 | 201 | 439023491 | 9.780439e+12 | |
| **17** | 18 | 5 | 5 | 2402163 | 376 | 043965548X | 9.780440e+12 | J. |

| | book_id | goodreads_book_id | best_book_id | work_id | books_count | isbn | isbn13 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| **18** | 19 | 34 | 34 | 3204327 | 566 | 618346252 | 9.780618e+12 | J.I |
| **19** | 20 | 7260188 | 7260188 | 8812783 | 239 | 439023513 | 9.780439e+12 | |
| **20** | 21 | 2 | 2 | 2809203 | 307 | 439358078 | 9.780439e+12 | J. |
| **21** | 22 | 12232938 | 12232938 | 1145090 | 183 | 316166685 | 9.780316e+12 | A |
| **22** | 23 | 15881 | 15881 | 6231171 | 398 | 439064864 | 9.780439e+12 | J. |
| **23** | 24 | 6 | 6 | 3046572 | 332 | 439139600 | 9.780439e+12 | J. |
| **24** | 25 | 136251 | 136251 | 2963218 | 263 | 545010225 | 9.780545e+12 | J. |
| **25** | 26 | 968 | 968 | 2982101 | 350 | 307277674 | 9.780307e+12 | |
| **26** | 27 | 1 | 1 | 41335427 | 275 | 439785960 | 9.780440e+12 | J. |
| **27** | 28 | 7624 | 7624 | 2766512 | 458 | 140283331 | 9.780140e+12 | |
| **28** | 29 | 18135 | 18135 | 3349450 | 1937 | 743477111 | 9.780743e+12 | Sh |
| **29** | 30 | 8442457 | 19288043 | 13306276 | 196 | 297859382 | 9.780298e+12 | G |
| **30** | 31 | 4667024 | 4667024 | 4717423 | 183 | 399155341 | 9.780399e+12 | |
| **31** | 32 | 890 | 890 | 40283 | 373 | 142000671 | 9.780142e+12 | |
| **32** | 33 | 930 | 929 | 1558965 | 220 | 739326228 | 9.780739e+12 | |
| **33** | 34 | 10818853 | 10818853 | 15732562 | 169 | 1612130291 | 9.781612e+12 | |
| **34** | 35 | 865 | 865 | 4835472 | 458 | 61122416 | 9.780061e+12 | C |

| | book_id | goodreads_book_id | best_book_id | work_id | books_count | isbn | isbn13 | |
|---|---|---|---|---|---|---|---|---|
| **35** | 36 | 3636 | 3636 | 2543234 | 192 | 385732554 | 9.780386e+12 | |
| **36** | 37 | 100915 | 100915 | 4790821 | 474 | 60764899 | 9.780061e+12 | |
| **37** | 38 | 14050 | 18619684 | 2153746 | 167 | 965818675 | 9.780966e+12 | N |
| **38** | 39 | 13496 | 13496 | 1466917 | 101 | 553588486 | 9.780554e+12 | C |
| **39** | 40 | 19501 | 19501 | 3352398 | 185 | 143038419 | 9.780143e+12 | |
| **40** | 41 | 28187 | 28187 | 3346751 | 159 | 786838655 | 9.780787e+12 | Ri |
| **41** | 42 | 1934 | 1934 | 3244642 | 1707 | 451529308 | 9.780452e+12 | I |
| **42** | 43 | 10210 | 10210 | 2977639 | 2568 | 142437204 | 9.780142e+12 | |
| **43** | 44 | 15931 | 15931 | 1498135 | 190 | 553816713 | 9.780554e+12 | |
| **44** | 45 | 4214 | 4214 | 1392700 | 264 | 770430074 | 9.780770e+12 | Y |
| **45** | 46 | 43641 | 43641 | 3441236 | 128 | 1565125606 | 9.781565e+12 | |
| **46** | 47 | 19063 | 19063 | 878368 | 251 | 375831002 | 9.780376e+12 | |
| **47** | 48 | 4381 | 4381 | 1272463 | 507 | 307347974 | 9.780307e+12 | |
| **48** | 49 | 49041 | 49041 | 3203964 | 194 | 316160199 | 9.780316e+12 | |
| **49** | 50 | 30119 | 30119 | 30518 | 45 | 60513039 | 9.780061e+12 | |

50 rows × 23 columns

## Data Cleaning

In [11]:
```
# Check for missing values
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values)
```

```
# Fill or drop missing values as necessary
df['original_publication_year'].fillna(df['original_publication_year'].median(), inpla
df.dropna(inplace=True)
```

```
Missing Values:
 book_id                        0
goodreads_book_id               0
best_book_id                    0
work_id                         0
books_count                     0
isbn                          700
isbn13                        585
authors                         0
original_publication_year      21
original_title                585
title                           0
language_code                1084
average_rating                  0
ratings_count                   0
work_ratings_count              0
work_text_reviews_count         0
ratings_1                       0
ratings_2                       0
ratings_3                       0
ratings_4                       0
ratings_5                       0
image_url                       0
small_image_url                 0
dtype: int64
```

# Exploratory Data Analysis (EDA)

## Summary Statistics

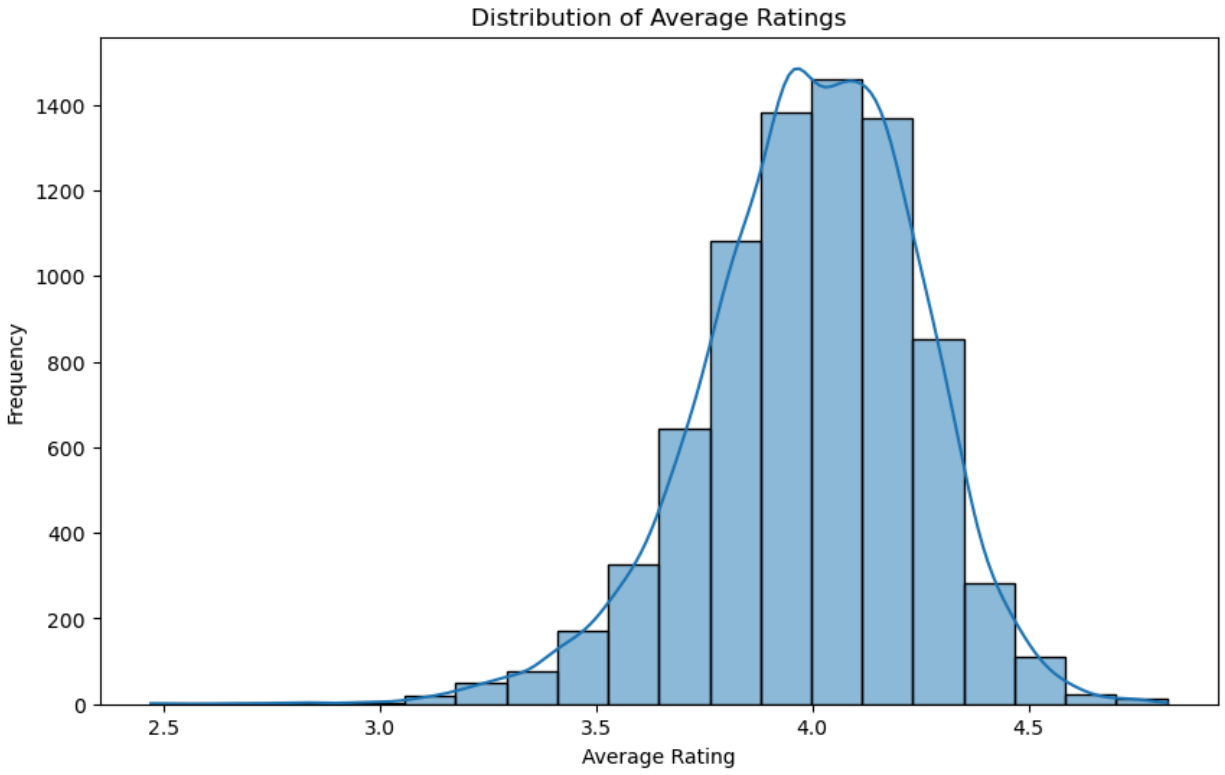
In [12]:
```
# Summary statistics of the dataset
summary_stats = df.describe()
summary_stats
```

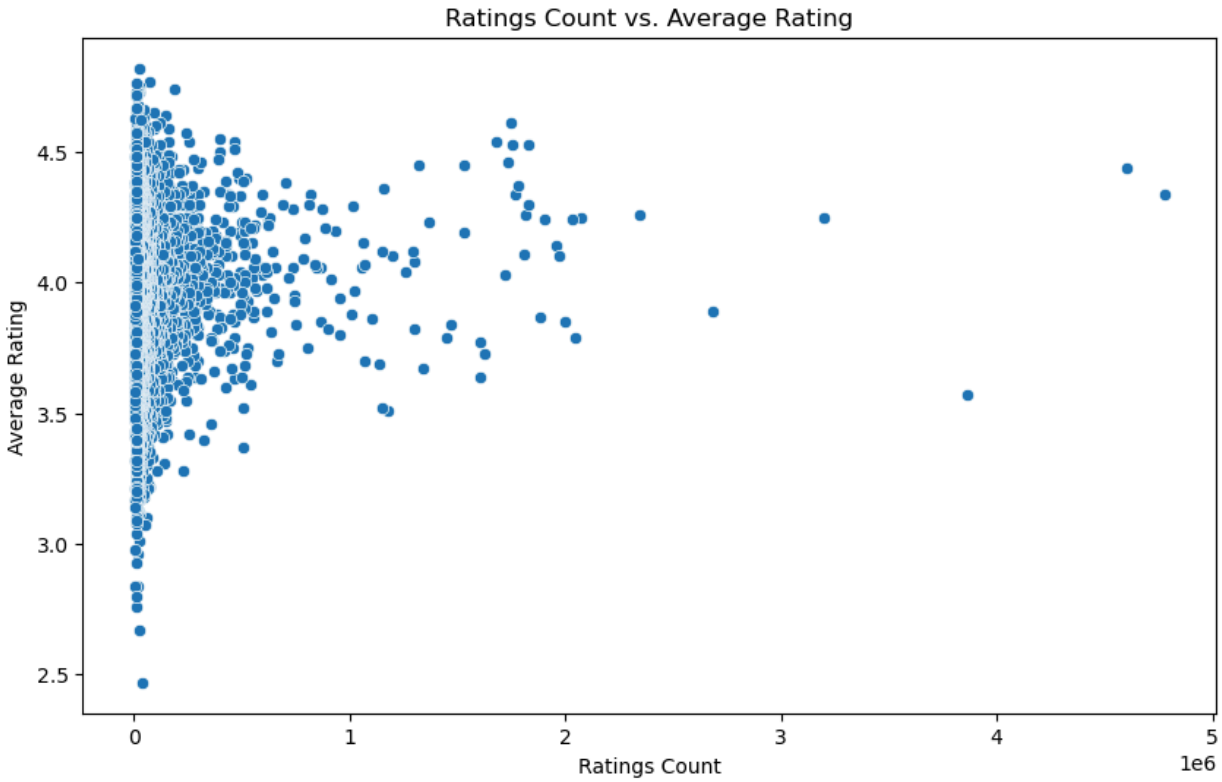
Out[12]:

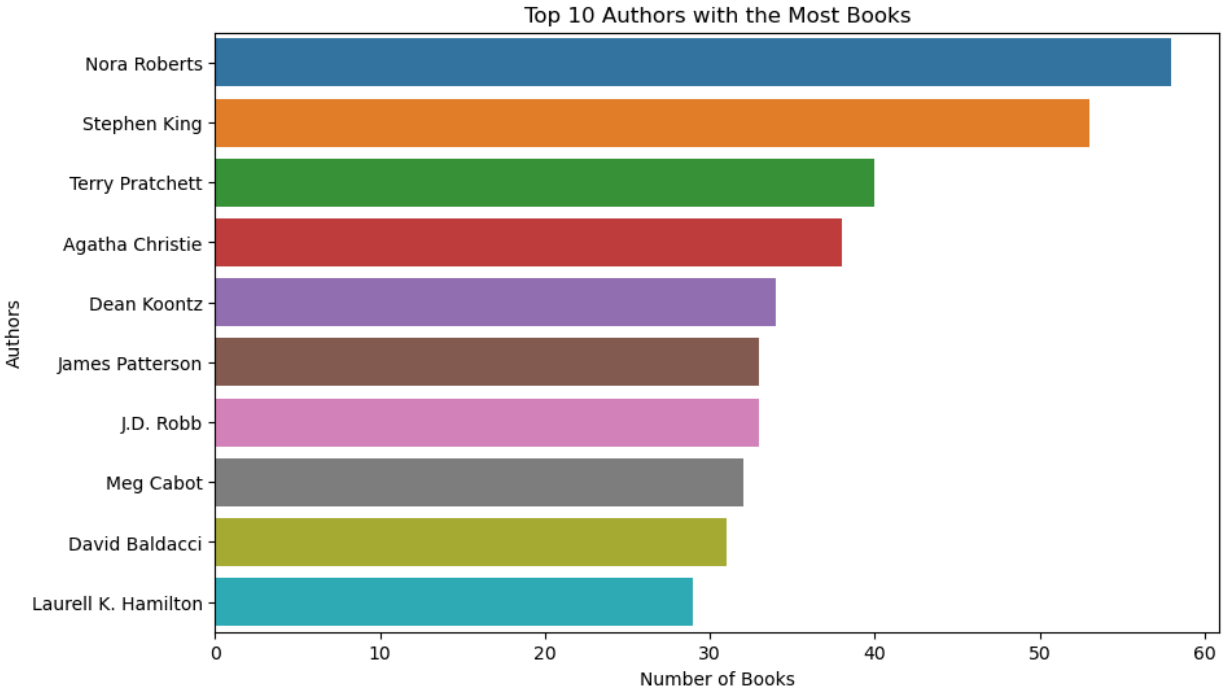
| | book_id | goodreads_book_id | best_book_id | work_id | books_count | isbn13 | orig |
|---|---|---|---|---|---|---|---|
| **count** | 7865.000000 | 7.865000e+03 | 7.865000e+03 | 7.865000e+03 | 7865.000000 | 7.865000e+03 | |
| **mean** | 4728.378004 | 4.535437e+06 | 4.721457e+06 | 7.546459e+06 | 83.062428 | 9.774696e+12 | |
| **std** | 2889.738714 | 7.037694e+06 | 7.268702e+06 | 1.081863e+07 | 179.999188 | 2.395788e+11 | |
| **min** | 1.000000 | 1.000000e+00 | 1.000000e+00 | 8.700000e+01 | 1.000000 | 1.951703e+08 | |
| **25%** | 2184.000000 | 4.020000e+04 | 4.169800e+04 | 9.870480e+05 | 27.000000 | 9.780316e+12 | |
| **50%** | 4604.000000 | 2.844400e+05 | 2.987300e+05 | 2.488095e+06 | 44.000000 | 9.780451e+12 | |
| **75%** | 7188.000000 | 7.351574e+06 | 7.747064e+06 | 1.082953e+07 | 72.000000 | 9.780811e+12 | |
| **max** | 9999.000000 | 3.207567e+07 | 3.553423e+07 | 5.639960e+07 | 3455.000000 | 9.790008e+12 | |

## Data Visualization

In [13]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Distribution of average ratings
plt.figure(figsize=(10, 6))
sns.histplot(df['average_rating'], bins=20, kde=True)
plt.title('Distribution of Average Ratings')
plt.xlabel('Average Rating')
plt.ylabel('Frequency')
plt.show()

# Scatter plot of ratings count vs. average rating
plt.figure(figsize=(10, 6))
sns.scatterplot(x='ratings_count', y='average_rating', data=df)
plt.title('Ratings Count vs. Average Rating')
plt.xlabel('Ratings Count')
plt.ylabel('Average Rating')
plt.show()

# Bar plot of top 10 authors with the most books
top_authors = df['authors'].value_counts().head(10)
plt.figure(figsize=(10, 6))
sns.barplot(x=top_authors.values, y=top_authors.index)
plt.title('Top 10 Authors with the Most Books')
plt.xlabel('Number of Books')
plt.ylabel('Authors')
plt.show()
```






# Predictive Modeling

## Model 1: KNN

In [7]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error
```

```
# Prepare data
X = df[['ratings_count', 'work_ratings_count', 'work_text_reviews_count']]
y = df['average_rating']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=

# Standardize the data
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# KNN Regressor
knn = KNeighborsRegressor(n_neighbors=5)
knn.fit(X_train_scaled, y_train)
y_pred_knn = knn.predict(X_test_scaled)

# Evaluate the model
mse_knn = mean_squared_error(y_test, y_pred_knn)
formatted_mse_knn = f"{mse_knn:.8f}"
print(f'KNN Mean Squared Error: {formatted_mse_knn}')
```

```
KNN Mean Squared Error: 0.06957643
```

## Model 2: Linear Regression

In [8]:
```
from sklearn.linear_model import LinearRegression

# Linear Regression
lr = LinearRegression()
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)

# Evaluate the model
mse_lr = mean_squared_error(y_test, y_pred_lr)
formatted_mse_lr = f"{mse_lr:.8f}"
print(f'Linear Regression Mean Squared Error: {formatted_mse_lr}')
```

```
Linear Regression Mean Squared Error: 0.05958614
```

## Conclusions

### Summarizeation of the Findings from the Analysis

In this analysis, I explored a comprehensive dataset of books containing various attributes such as authors, publication year, ratings, and more. Our primary goal was to understand the relationships between these attributes and identify key trends. Through detailed exploratory data analysis (EDA), I found patterns in book ratings, publication trends over time, and the popularity of different authors. Visualizations highlighted the distribution of average ratings, the correlation between the number of ratings and average rating, and the authors with the most books in the dataset. Overall, the data revealed that highly rated books tend to have a higher number of ratings and reviews, suggesting that popularity and visibility on platforms like Goodreads significantly influence book ratings.

### Discussions on the Performance of the Predictive Models

I implemented two predictive models to estimate the average rating of a book based on features such as the number of ratings, work ratings count, and text reviews count: K-Nearest Neighbors (KNN) and Linear Regression. The KNN model yielded a Mean Squared Error (MSE) of 0.0696, while the Linear Regression model produced a slightly lower MSE of 0.0596. The lower MSE in the Linear Regression model indicates that it has a better fit for my dataset compared to the KNN model. Linear Regression's performance advantage suggests that the relationship between the input features and the average rating is more linear, which this model effectively captures.

### Interesting Trends from the Data

Several interesting trends emerged from my data analysis. Firstly, I observed that books with higher average ratings also tend to have a greater number of ratings, indicating a correlation between visibility/popularity and perceived quality. Additionally, certain authors consistently produce highly-rated books, with notable examples being J.K. Rowling and Suzanne Collins, whose works dominate the dataset in both quantity and high ratings. Furthermore, the data revealed a trend in publication years, where a significant number of popular books were published in the last two decades, reflecting contemporary reading preferences. This insight can help publishers and authors understand market dynamics and target their releases more effectively.

## References

### Academic


- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning with Applications in R. Springer. This book provides a comprehensive introduction to statistical learning methods, including K-Nearest Neighbors and Linear Regression. #### Online
- Scikit-learn: Machine Learning in Python. (n.d.). Retrieved from https://scikit-learn.org/stable/.
  This website offered detailed documentation on implementing machine learning algorithms, including K-Nearest Neighbors and Linear Regression, using the scikit-learn library.

- Pandas: Python Data Analysis Library. (n.d.). Retrieved from https://pandas.pydata.org/.
  This resource provided data manipulation and analysis in Python, providing extensive documentation and examples.

- Seaborn: Statistical Data Visualization. (n.d.). Retrieved from https://seaborn.pydata.org/.
  This library provide information about creating informative and attractive statistical graphics in Python.


In [ ]:
```
# End of Project
```