1. Domain-specific area and objectives of the project (200-500 words)

"Today a reader, tomorrow a leader.", is a famous quote by Margrett Fuller, a transcendentalist and a full-time book reviewer in Journalism. The quote showcases the importance of reading and how powerful it is in helping shape future leaders of the world. Reading is not only a hobby but also a habit that must be picked up at an early age to encourage people to continue a lifelong journey of learning and development. However, a survey conducted by the National Endowment For The Arts(NEA) in 2022 found a decrease in the number of adults who read more than one book for leisure. The number fell to 48.5% down from 52.7% in 2017(***Milliot, Nea finds worrying drop in reading participation,2023***). This decline in reading literacy among adults shows concern as the decrease in the number of readers in 2022 is greater than the decline in the survey conducted in 2012 and 2017, with a gap of only two percent. This shows how more adults are reading less which in turn hurts book sales and is a cause of concern among bookstore owners. How then do bookstores attract readers and boost their sales?. Well, bookstores need to analyze the market trends on which books of various authors and genres are preferred by readers. Then, we can identify readers' interests over time, and make sound decisions on the type of books to keep in stock to attract more readers. This is where my linear regression model is put to use. My model can analyze and predict the trends in readers' preferences and predict the number of books of the highest-grossing author and genre to be kept in stock to attract more customers to the store. This will allow stores to generate more profit by making data-driven decisions on which popular books to purchase and sell.

2. Dataset description (200-500 words)

The dataset I will be using is a combination of two CSV files obtained from the official Kaggle website. The first CSV file is titled “Books Sales and Ratings”. The original author of this dataset is Josh Murrey. This CSV file contains fifteen columns and consists of only Text and Numeric data types. The dataset has a size of 16050 entries. The second CSV file is titled “best-selling-books”. The CSV file is owned by D Rahulsingh and is gathered from original and independent sources. This CSV file contains six columns and consists mainly of strings and integers as data types. This dataset has a size of only 1044 entries. Both CSV files will be merged and only entries from the columns belonging to attributes Author, Genre, sales, and units sold will be used to build and train the machine learning model after data preparation. Hence, the size of the new book dataset will be kept to less than 10,000 entries in total. This will allow the dataset to load and process quickly. The new dataset will also be processed further to handle any missing values. It will be converted to the appropriate structure before being used to train and test the machine-learning model. The dataset will also be split into two sections. One for Training and the other for Testing using the train_test_split feature that allows us to split arrays into subsets that can be used for validation of the new dataset.

3. Data preparation (acquisition/cleaning/sanitisation/normalisation)

In [2]:
```
#import all libraries here
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
```

```
from sklearn.model_selection import KFold
from sklearn import metrics
from sklearn.metrics import mean_squared_error



#Load the two CSV files
Book_data_1 = pd.read_csv("Books_Data_Clean.csv")
Book_data_2 = pd.read_csv("best-selling-books.csv")

#Merge the two CSV files using the concat method
New_book_data = pd.concat(map(pd.read_csv, ['Books_Data_Clean.csv', 'best-selling-books.cs
print(New_book_data)
```

```
      index  Publishing Year                        Book Name  \
0       0.0           1975.0                          Beowulf
1       1.0           1987.0                Batman: Year One
2       2.0           2015.0               Go Set a Watchman
3       3.0           2008.0  When You Are Engulfed in Flames
4       4.0           2011.0        Daughter of Smoke & Bone
...     ...              ...                              ...
1239    NaN              NaN                              NaN
1240    NaN              NaN                              NaN
1241    NaN              NaN                              NaN
1242    NaN              NaN                              NaN
1243    NaN              NaN                              NaN

                                              Author language_code  \
0                                Unknown, Seamus Heaney         en-US
1     Frank Miller, David Mazzucchelli, Richmond Lew...           eng
2                                          Harper Lee            eng
3                                       David Sedaris         en-US
4                                        Laini Taylor            eng
...                                               ...           ...
1239                                             NaN           NaN
1240                                             NaN           NaN
1241                                             NaN           NaN
1242                                             NaN           NaN
1243                                             NaN           NaN

     Author_Rating  Book_average_rating  Book_ratings_count          genre  \
0           Novice                3.42            155903.0  genre fiction
1     Intermediate                4.23            145267.0  genre fiction
2           Novice                3.31            138669.0  genre fiction
3     Intermediate                4.04            150898.0        fiction
4     Intermediate                4.04            198283.0  genre fiction
...            ...                 ...                 ...            ...
1239           NaN                 NaN                 NaN            NaN
1240           NaN                 NaN                 NaN            NaN
1241           NaN                 NaN                 NaN            NaN
1242           NaN                 NaN                 NaN            NaN
1243           NaN                 NaN                 NaN            NaN

      gross sales  ...  sale price  sales rank  \
0         34160.0  ...        4.88         1.0
1         12437.5  ...        1.99         2.0
2         47795.0  ...        8.69         3.0
3         41250.0  ...        7.50         3.0
4         37952.5  ...        7.99         4.0
...           ...  ...         ...         ...
1239          NaN  ...         NaN         NaN
1240          NaN  ...         NaN         NaN
1241          NaN  ...         NaN         NaN
1242          NaN  ...         NaN         NaN
1243          NaN  ...         NaN         NaN
```

```
                          Publisher  units sold  \
0           HarperCollins Publishers     7000.0
1           HarperCollins Publishers     6250.0
2     Amazon Digital Services,  Inc.     5500.0
3                Hachette Book Group     5500.0
4             Penguin Group (USA) LLC     4750.0
...                               ...        ...
1239                              NaN        NaN
1240                              NaN        NaN
1241                              NaN        NaN
1242                              NaN        NaN
1243                              NaN        NaN

                                                  Book  \
0                                                  NaN
1                                                  NaN
2                                                  NaN
3                                                  NaN
4                                                  NaN
...                                                ...
1239                                          The Goal
1240                                     Fahrenheit 451
1241                                     Angela's Ashes
1242  The Story of My Experiments with Truth (સત્યનાં...
1243                               Bridget Jones's Diary

                       Author(s) Original language First published  \
0                            NaN               NaN             NaN
1                            NaN               NaN             NaN
2                            NaN               NaN             NaN
3                            NaN               NaN             NaN
4                            NaN               NaN             NaN
...                          ...               ...             ...
1239         Eliyahu M. Goldratt           English          1984.0
1240               Ray Bradbury           English          1953.0
1241              Frank McCourt           English          1996.0
1242  Mohandas Karamchand Gandhi          Gujarati          1929.0
1243              Helen Fielding           English          1996.0

      Approximate sales in millions  Genre
0                               NaN    NaN
1                               NaN    NaN
2                               NaN    NaN
3                               NaN    NaN
4                               NaN    NaN
...                             ...    ...
1239                           10.0    NaN
1240                           10.0    NaN
1241                           10.0    NaN
1242                           10.0    NaN
1243                           10.0    NaN

[1244 rows x 21 columns]
```

In [4]:

```
#Join the rows that have the same attributes such as Author, Genre, book name, gross sales
new_joined_dataset = pd.merge(Book_data_1,Book_data_2, how='left', left_on=['Book Name','A
new_joined_dataset

#drop columns that are unnecessary
new_clean_dataset = new_joined_dataset.drop(['Book', 'Author(s)', "Genre", 'First publishe
new_clean_dataset = pd.DataFrame(new_clean_dataset)
new_clean_dataset
```

Out[4]:

	index	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_
0	0	1975.0	Beowulf	Unknown, Seamus Heaney	en-US	Novice	3.42	1
1	1	1987.0	Batman: Year One	Frank Miller, David Mazzucchelli, Richmond Lew...	eng	Intermediate	4.23	1
2	2	2015.0	Go Set a Watchman	Harper Lee	eng	Novice	3.31	1
3	3	2008.0	When You Are Engulfed in Flames	David Sedaris	en-US	Intermediate	4.04	1
4	4	2011.0	Daughter of Smoke & Bone	Laini Taylor	eng	Intermediate	4.04	1
...	...	...	...	...	...	...	...	
1065	1065	2014.0	Gray Mountain	John Grisham	eng	Intermediate	3.52	
1066	1066	1989.0	The Power of One	Bryce Courtenay	eng	Excellent	4.34	
1067	1067	1930.0	The Maltese Falcon	Dashiell Hammett	eng	Intermediate	3.92	
1068	1068	2011.0	Night Road	Kristin Hannah	en-US	Excellent	4.17	
1069	1069	1999.0	Tripwire	Lee Child	eng	Excellent	4.07	

1070 rows × 15 columns

In [5]:
```
#Identify if the dataset has any missing values in columns
new_clean_dataset.isnull().all()
```

Out[5]:
```
index                 False
Publishing Year       False
Book Name             False
Author                False
language_code         False
Author_Rating         False
Book_average_rating   False
Book_ratings_count    False
genre                 False
gross sales           False
publisher revenue     False
sale price            False
sales rank            False
Publisher             False
```

```
units sold             False
dtype: bool
```

In [6]:
```
#Counting the NaN in each column
new_clean_dataset.isnull().sum()
```

Out[6]:
```
index                   0
Publishing Year         1
Book Name              23
Author                  0
language_code          53
Author_Rating           0
Book_average_rating     0
Book_ratings_count      0
genre                   0
gross sales             0
publisher revenue       0
sale price              0
sales rank              0
Publisher               0
units sold              0
dtype: int64
```

In [7]:
```
#Identify if the dataset has any missing values in rows.
new_clean_dataset.isnull().all(axis=1)
```

Out[7]:
```
0       False
1       False
2       False
3       False
4       False
        ...
1065    False
1066    False
1067    False
1068    False
1069    False
Length: 1070, dtype: bool
```

In [8]:
```
#Counting the NaN in each row
new_clean_dataset.isnull().sum(axis=1)
```

Out[8]:
```
0       0
1       0
2       0
3       0
4       0
       ..
1065    0
1066    0
1067    0
1068    0
1069    0
Length: 1070, dtype: int64
```

In [9]:
```
#Counting the total number of NaN values in columns
print(new_clean_dataset.isnull().values.sum())
```

```
77
```

In [10]:
```
#Fill the NaN value with a median value for the publishing year.
Publishing_Year_median = new_clean_dataset['Publishing Year'].median()
```

```
new_clean_dataset['Publishing Year'] = new_clean_dataset['Publishing Year'].fillna(Publish
new_clean_dataset.head()
```

Out[10]:

	index	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_cou
0	0	1975.0	Beowulf	Unknown, Seamus Heaney	en-US	Novice	3.42	1559
1	1	1987.0	Batman: Year One	Frank Miller, David Mazzucchelli, Richmond Lew...	eng	Intermediate	4.23	1452
2	2	2015.0	Go Set a Watchman	Harper Lee	eng	Novice	3.31	1386
3	3	2008.0	When You Are Engulfed in Flames	David Sedaris	en-US	Intermediate	4.04	1508
4	4	2011.0	Daughter of Smoke & Bone	Laini Taylor	eng	Intermediate	4.04	1982

In [11]:
```
#use the sum feature to view if the NaN value for the publishing year has been successful
new_clean_dataset.isnull().sum()
```

Out[11]:
```
index                  0
Publishing Year        0
Book Name             23
Author                 0
language_code         53
Author_Rating          0
Book_average_rating    0
Book_ratings_count     0
genre                  0
gross sales            0
publisher revenue      0
sale price             0
sales rank             0
Publisher              0
units sold             0
dtype: int64
```

In [12]:
```
#Drop the NaN values in the Book Name column.
new_clean_dataset.dropna(subset=['Book Name'], inplace=True)
new_clean_dataset.head()
```

Out[12]:

	index	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_cou
0	0	1975.0	Beowulf	Unknown, Seamus Heaney	en-US	Novice	3.42	1559

	index	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_cou
1	1	1987.0	Batman: Year One	Frank Miller, David Mazzucchelli, Richmond Lew...	eng	Intermediate	4.23	1452
2	2	2015.0	Go Set a Watchman	Harper Lee	eng	Novice	3.31	1386
3	3	2008.0	When You Are Engulfed in Flames	David Sedaris	en-US	Intermediate	4.04	1508
4	4	2011.0	Daughter of Smoke & Bone	Laini Taylor	eng	Intermediate	4.04	1982

In [13]:
```
#use the sum feature to view if the NaN value for Book Names has been successfully replace
new_clean_dataset.isnull().sum()
```

Out[13]:
```
index                  0
Publishing Year        0
Book Name              0
Author                 0
language_code         49
Author_Rating          0
Book_average_rating    0
Book_ratings_count     0
genre                  0
gross sales            0
publisher revenue      0
sale price             0
sales rank             0
Publisher              0
units sold             0
dtype: int64
```

In [14]:
```
#Drop the NaN values in the language code column.
new_clean_dataset.dropna(subset=['language_code'], inplace=True)
new_clean_dataset.head()
```

Out[14]:

	index	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_cou
0	0	1975.0	Beowulf	Unknown, Seamus Heaney	en-US	Novice	3.42	1559
1	1	1987.0	Batman: Year One	Frank Miller, David Mazzucchelli, Richmond Lew...	eng	Intermediate	4.23	1452
2	2	2015.0	Go Set a Watchman	Harper Lee	eng	Novice	3.31	1386

	index	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_cou
3	3	2008.0	When You Are Engulfed in Flames	David Sedaris	en-US	Intermediate	4.04	1508
4	4	2011.0	Daughter of Smoke & Bone	Laini Taylor	eng	Intermediate	4.04	1982

In [15]:
```
#use the sum feature to view if the NaN value for language code has been successfully repl
new_clean_dataset.isnull().sum()
```

Out[15]:
```
index                0
Publishing Year      0
Book Name            0
Author               0
language_code        0
Author_Rating        0
Book_average_rating  0
Book_ratings_count   0
genre                0
gross sales          0
publisher revenue    0
sale price           0
sales rank           0
Publisher            0
units sold           0
dtype: int64
```

In [16]:
```
#view the features in the new dataset after replacing NaN values
new_clean_dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 998 entries, 0 to 1069
Data columns (total 15 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   index                998 non-null    int64  
 1   Publishing Year      998 non-null    float64
 2   Book Name            998 non-null    object 
 3   Author               998 non-null    object 
 4   language_code        998 non-null    object 
 5   Author_Rating        998 non-null    object 
 6   Book_average_rating  998 non-null    float64
 7   Book_ratings_count   998 non-null    int64  
 8   genre                998 non-null    object 
 9   gross sales          998 non-null    float64
 10  publisher revenue    998 non-null    float64
 11  sale price           998 non-null    float64
 12  sales rank           998 non-null    int64  
 13  Publisher            998 non-null    object 
 14  units sold           998 non-null    int64  
dtypes: float64(5), int64(4), object(6)
memory usage: 124.8+ KB
```

In [17]:
```
#Convert Publishing Year from float to integer
new_clean_dataset['Publishing Year'] = new_clean_dataset['Publishing Year'].astype(int)
#display the datatype for the Publishing Year after conversion
display(new_clean_dataset.dtypes)
```

```
index                   int64
Publishing Year         int32
Book Name              object
Author                 object
language_code          object
Author_Rating          object
Book_average_rating   float64
Book_ratings_count      int64
genre                  object
gross sales           float64
publisher revenue     float64
sale price            float64
sales rank              int64
Publisher              object
units sold              int64
dtype: object
```

In [18]:

```
#drop the index and publisher revenue.
new_dataset_1 = new_clean_dataset.drop(['index', 'publisher revenue'], axis=1)
new_dataset_1.head()
```

Out[18]:

	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_count	ger
0	1975	Beowulf	Unknown, Seamus Heaney	en-US	Novice	3.42	155903	ger ficti
1	1987	Batman: Year One	Frank Miller, David Mazzucchelli, Richmond Lew...	eng	Intermediate	4.23	145267	ger ficti
2	2015	Go Set a Watchman	Harper Lee	eng	Novice	3.31	138669	ger ficti
3	2008	When You Are Engulfed in Flames	David Sedaris	en-US	Intermediate	4.04	150898	ficti
4	2011	Daughter of Smoke & Bone	Laini Taylor	eng	Intermediate	4.04	198283	ger ficti

In [19]:

```
#Converting the Author to 1NF
new_dataset_1['Author']=new_dataset_1['Author'].str.split(";")
new_dataset_1.explode(['Author'])
```

Out[19]:

	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_count
0	1975	Beowulf	Unknown, Seamus Heaney	en-US	Novice	3.42	155903
1	1987	Batman: Year One	Frank Miller, David Mazzucchelli, Richmond Lew...	eng	Intermediate	4.23	145267

	Publishing Year	Book Name	Author	language_code	Author_Rating	Book_average_rating	Book_ratings_count
2	2015	Go Set a Watchman	Harper Lee	eng	Novice	3.31	138669
3	2008	When You Are Engulfed in Flames	David Sedaris	en-US	Intermediate	4.04	150898
4	2011	Daughter of Smoke & Bone	Laini Taylor	eng	Intermediate	4.04	198283
...	...	...	...	...	...	...	...
1065	2014	Gray Mountain	John Grisham	eng	Intermediate	3.52	37379
1066	1989	The Power of One	Bryce Courtenay	eng	Excellent	4.34	57312
1067	1930	The Maltese Falcon	Dashiell Hammett	eng	Intermediate	3.92	58742
1068	2011	Night Road	Kristin Hannah	en-US	Excellent	4.17	58028
1069	1999	Tripwire	Lee Child	eng	Excellent	4.07	55251

998 rows × 13 columns

In [20]:
```
#save the dataset locally
new_dataset_1.to_csv('new_book_dataset')
```

4. Statistical analysis

In [21]:
```
#Measures of Central Tendency
#Mean, Median, Mode
#Calculating the mean gross sales
new_book_dataset = pd.read_csv('new_book_dataset')
new_book_dataset_df = pd.DataFrame(data=new_book_dataset)
Mean_gross_sales = new_book_dataset['gross sales'].mean()
print(Mean_gross_sales)
```

```
1885.0851503006031
```

In [22]:
```
#Calculating the median gross sales
Median_gross_sales = new_book_dataset['gross sales'].median()
print(Median_gross_sales)
```

```
806.25
```

In [23]:
```
#Calculating the mode for gross sales
```

```
Mode_gross_sales = new_book_dataset['gross sales'].mode()
print(Mode_gross_sales)
```

```
0    117.81
dtype: float64
```

In [24]:
```
#Calculating the mean units of books sold
Mean_units_books_sold = new_book_dataset['units sold'].mean()
print(Mean_units_books_sold)
```

```
9802.3126252505
```

In [25]:
```
#Calculating the median units of books sold
Median_units_books_sold = new_book_dataset['units sold'].median()
print(Median_units_books_sold)
```

```
3915.0
```

In [26]:
```
#Calculating the mode for the units of books sold
Mode_units_books_sold = new_book_dataset['units sold'].mode()
print(Mode_units_books_sold)
```

```
0     108
1    2916
dtype: int64
```

In [27]:
```
#Calculating the average book rating for each Author rating
Average_book_rating_for_Author_rating = new_book_dataset_df.groupby('Author_Rating')['Book
print(Average_book_rating_for_Author_rating)
```

```
Author_Rating
Excellent       4.165826
Famous          4.438936
Intermediate    3.907334
Novice          3.371724
Name: Book_average_rating, dtype: float64
```

In [28]:
```
#Measures of spread
#Range, Standard Deviation,Variance,Quartiles
#Range of spread of the gross sales
def Range_gross_sales(new_book_dataset_df):
    return  new_book_dataset_df.max() - new_book_dataset_df.min()
print(new_book_dataset_df[['gross sales']].agg(Range_gross_sales))
```

```
gross sales    47690.06
dtype: float64
```

In [29]:
```
#Standard deviation of the gross sales
print(new_book_dataset_df[['gross sales']].agg(['std']))
```

```
     gross sales
std   4023.26877
```

In [30]:
```
#Variance of the gross sales
Variance_gross_sales = new_book_dataset_df['gross sales'].var()
print(Variance_gross_sales)
```

```
16186691.593936529
```

In [31]:
```
#Quartiles of the gross sales
```

```
Quartiles_gross_sales = new_book_dataset_df['gross sales'].quantile([0.25,0.5,0.75])
print(Quartiles_gross_sales)
```

```
0.25     370.8825
0.50     806.2500
0.75    1492.9650
Name: gross sales, dtype: float64
```

In [32]:

```
#Range of spread of units of books sold
def Range_units_books_sold(new_book_dataset_df):
    return  new_book_dataset_df.max() - new_book_dataset_df.min()
print(new_book_dataset_df[['units sold']].agg(Range_units_books_sold))
```

```
units sold    61454
dtype: int64
```

In [33]:

```
#Standard deviation of units of books sold
print(new_book_dataset_df[['units sold']].agg(['std']))
```

```
       units sold
std  15503.088302
```

In [34]:

```
#Variance of units of books sold
Variance_units_books_sold = new_book_dataset_df['units sold'].var()
print(Variance_units_books_sold)
```

```
240345746.8851158
```

In [35]:

```
#Quartiles of units of books sold
Quartiles_units_books_sold = new_book_dataset_df['units sold'].quantile([0.25,0.5,0.75])
print(Quartiles_units_books_sold)
```

```
0.25     555.00
0.50    3915.00
0.75    5420.25
Name: units sold, dtype: float64
```

In [36]:

```
#Type of Distribution
#Discrete or continuous data
#Identifying the distribution for gross sales
Gross_sales_distribution = new_book_dataset_df['gross sales'].value_counts().describe()
print(Gross_sales_distribution)
```

```
count    775.000000
mean       1.287742
std        0.942009
min        1.000000
25%        1.000000
50%        1.000000
75%        1.000000
max        9.000000
Name: gross sales, dtype: float64
```

In [37]:

```
#Identifying the distribution of units of books sold
Books_units_sold_distribution = new_book_dataset_df['units sold'].value_counts().describe
print(Books_units_sold_distribution)
```

```
count    478.000000
mean       2.087866
std        3.316407
min        1.000000
```

```
25%         1.000000
50%         1.000000
75%         2.000000
max        31.000000
Name: units sold, dtype: float64
```

5. Visualisation

Which factors affect the gross sales and the units of books sold?

Which Author/Authors and Genre are most popular?

Which year had the highest number of books read in which genre?

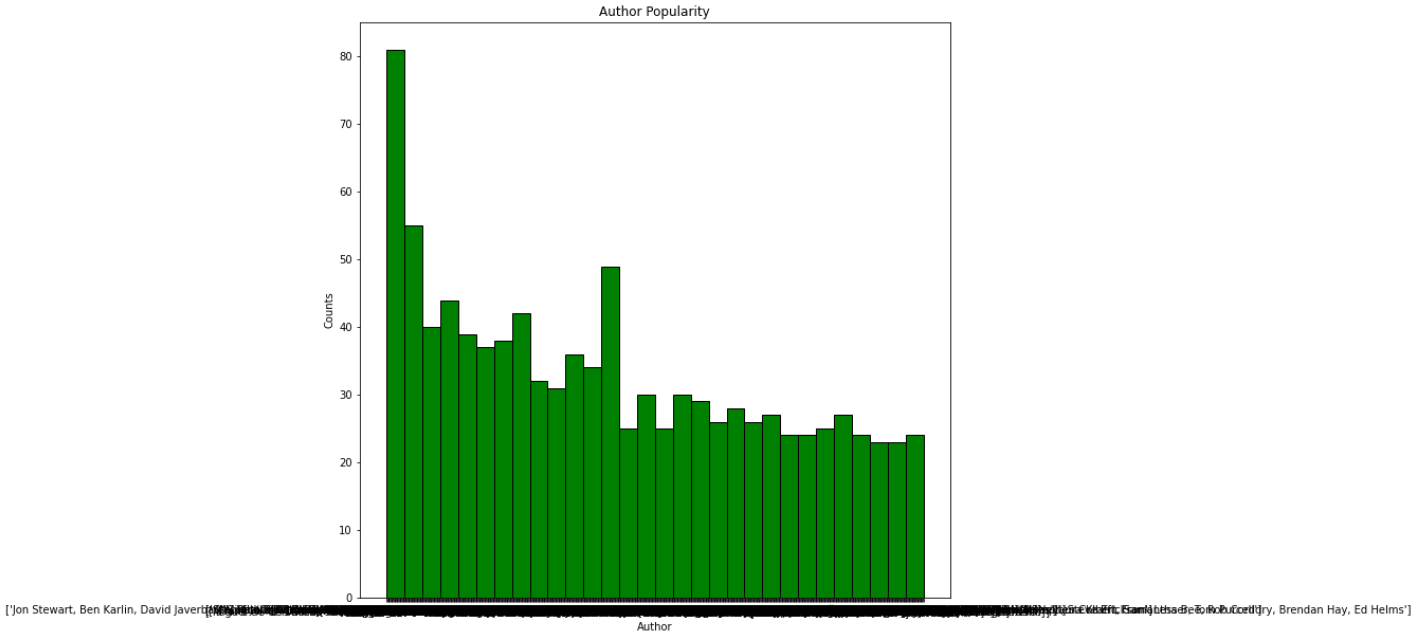
Factors that affect gross sales: Book_rating_count,Author_Rating, Book_average_rating, Sales price, genre

Factors that affect units of books sold: Book_rating_count, Author_Rating, Book_average_rating, genre

In [38]:
```
#Which Author/Authors most popular?
# Plot the histogram
plt.figure(figsize=(10,10))
Genre_Popularity = plt.hist(new_book_dataset_df['Author'], bins=30, color='Green', edgecol

# The title and the x and y axis labels are added
plt.xlabel('Author')
plt.ylabel('Counts')
plt.title('Author Popularity')

# The Histogram is displayed
plt.show()
```



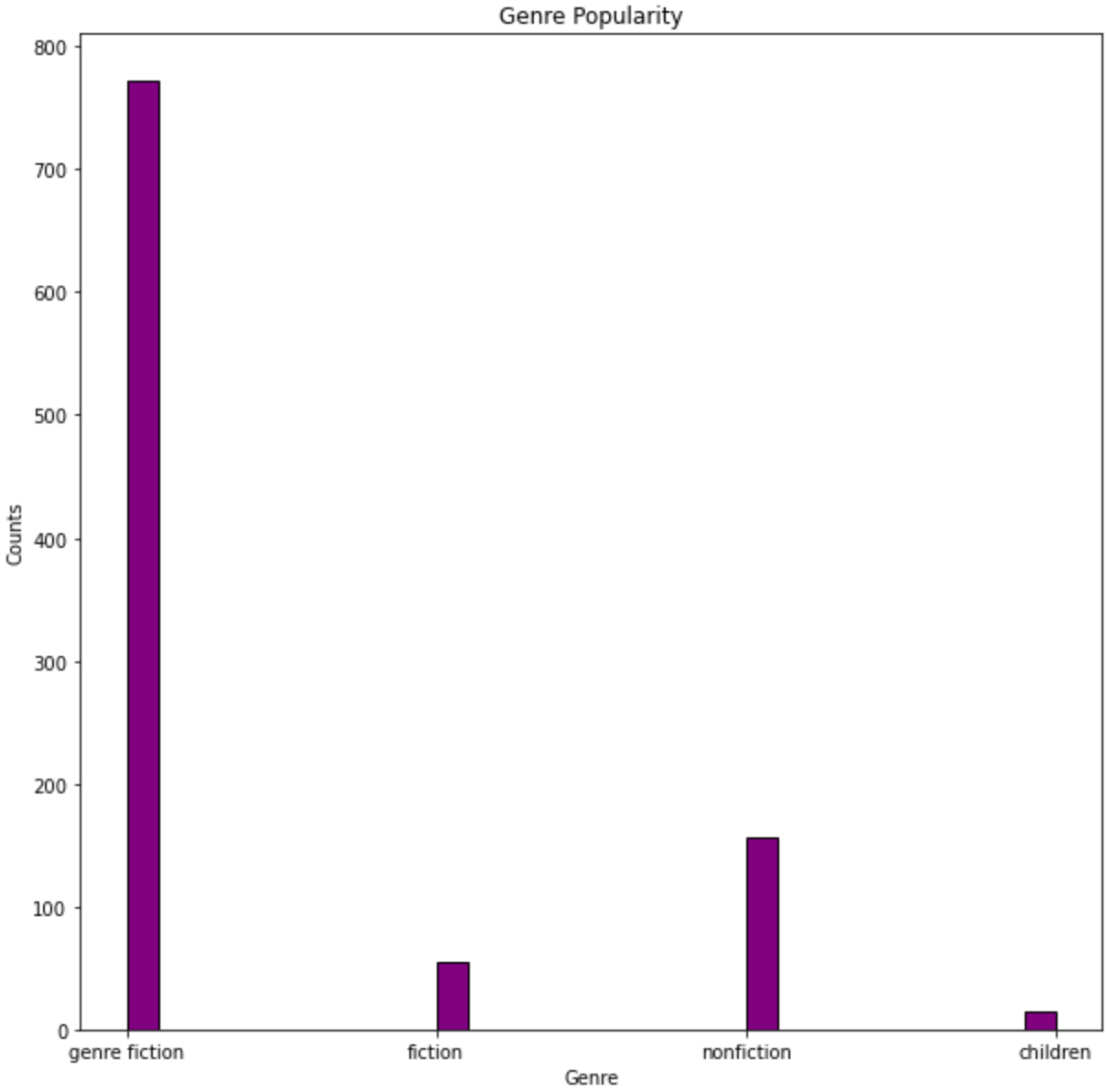
As seen from the histogram above, more than one author seems to have the highest number of books sold. Hence, it is harder to predict which Author is the most popular among readers over the years.

In [39]:
```
#Which Genre is the most popular?
# Plot the histogram
plt.figure(figsize=(10,10))
Genre_Popularity = plt.hist(new_book_dataset_df['genre'], bins=30, color='Purple', edgecol

# The title and the x and y axis labels are added
```

```
plt.xlabel('Genre')
plt.ylabel('Counts')
plt.title('Genre Popularity')

# The Histogram is displayed
plt.show()
```



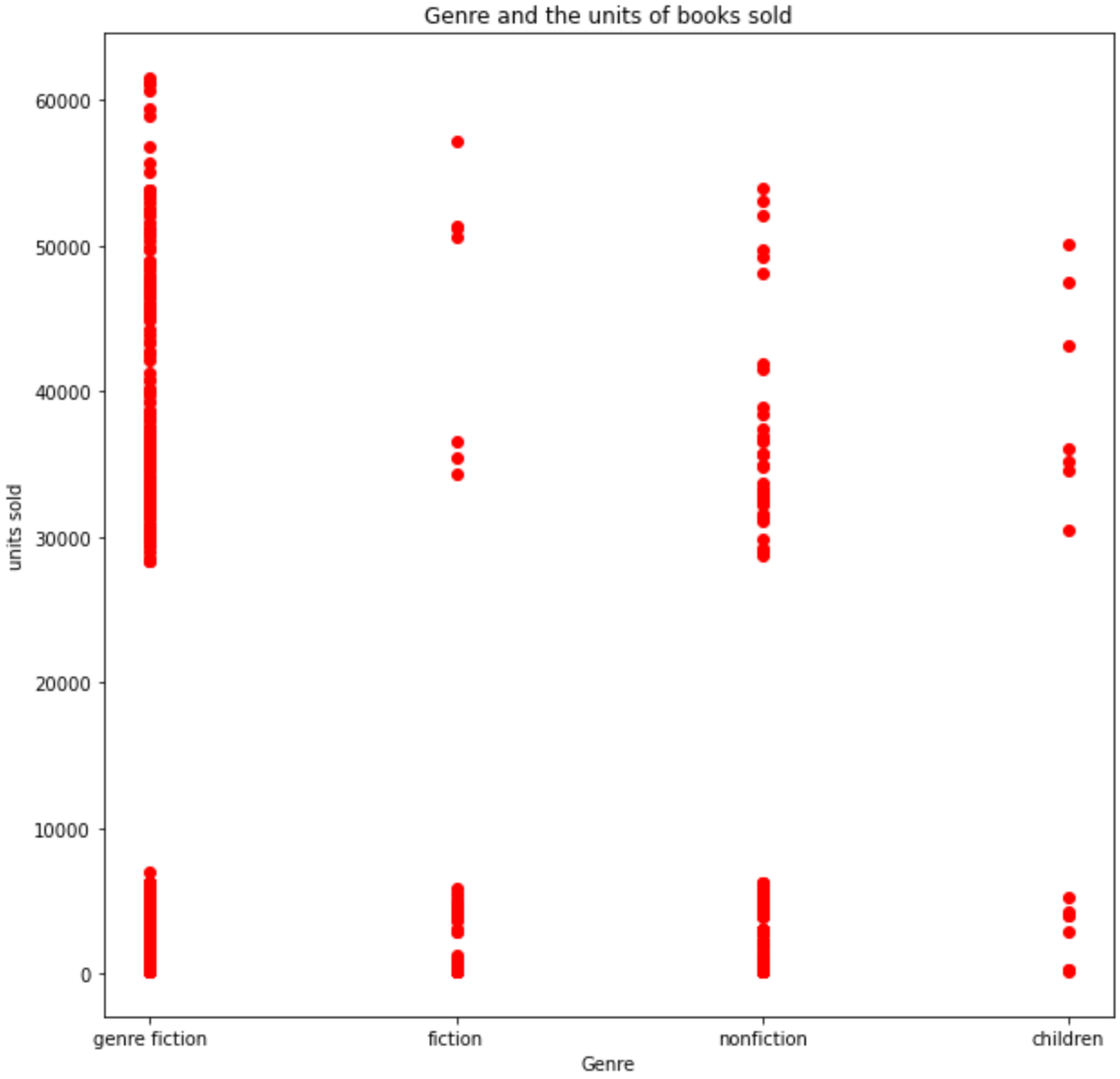
As seen from the histogram above, the genre fiction seems to have the highest number of books sold and it is the most popular genre among readers over the years.

In [40]:

```
#Which year had the highest number of books read in which genre?

#Firstly, we will compare the most popular genre, Fiction has how many units of books are
plt.figure(figsize=(10,10))
#Assign values to the x and y axis
x = new_book_dataset_df['genre']
y = new_book_dataset_df['units sold']
#Plot a scatter plot showcasing the genre of books and the units of books sold
Popular_genre_in_year = plt.scatter(x, y, color='red')
#label the axis and the title
plt.xlabel('Genre')
plt.ylabel('units sold')
plt.title('Genre and the units of books sold')
print(Popular_genre_in_year)
```

```
<matplotlib.collections.PathCollection object at 0x000002B611466B50>
```



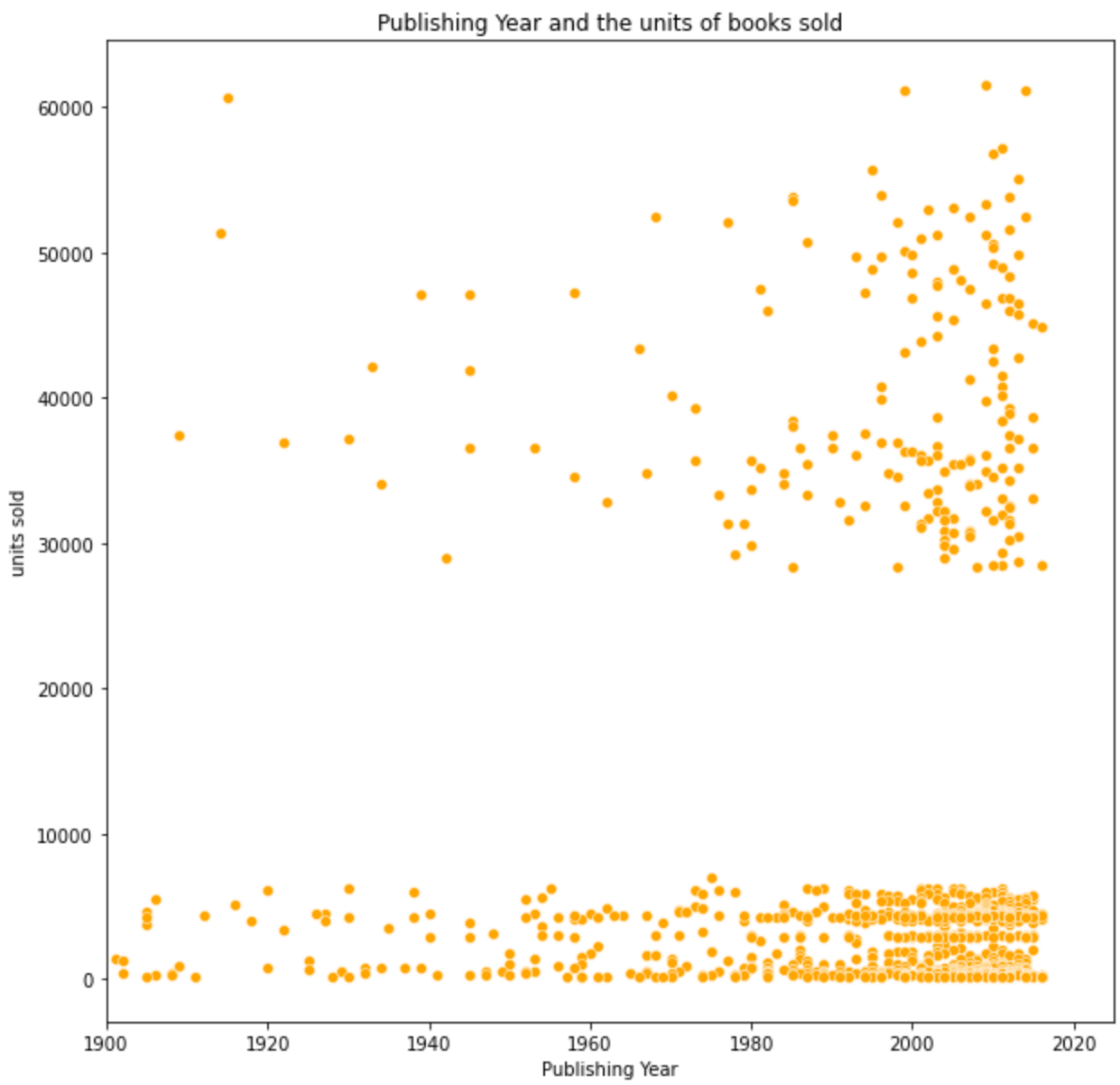
The scatter plot above shows that the genre fiction seems to have the highest number of books sold as it surpasses even 6000 units of books being sold over the year. The non-fiction genre is the second-highest genre that generated the most number of books sold.

In [41]:

```
#Now, we will look at the relationship between the Publishing Year and the number of books

# Plot the scatter plot
plt.figure(figsize=(10,10))
Publishing_year_units_sold = sns.scatterplot(data = new_book_dataset_df, x='Publishing Yea
Publishing_year_units_sold.set_xlim(1900,2025)
# The title and the x and y-axis labels are added
plt.xlabel('Publishing Year')
plt.ylabel('units sold')
plt.title('Publishing Year and the units of books sold')

# The scatter plot is displayed
plt.show()
```

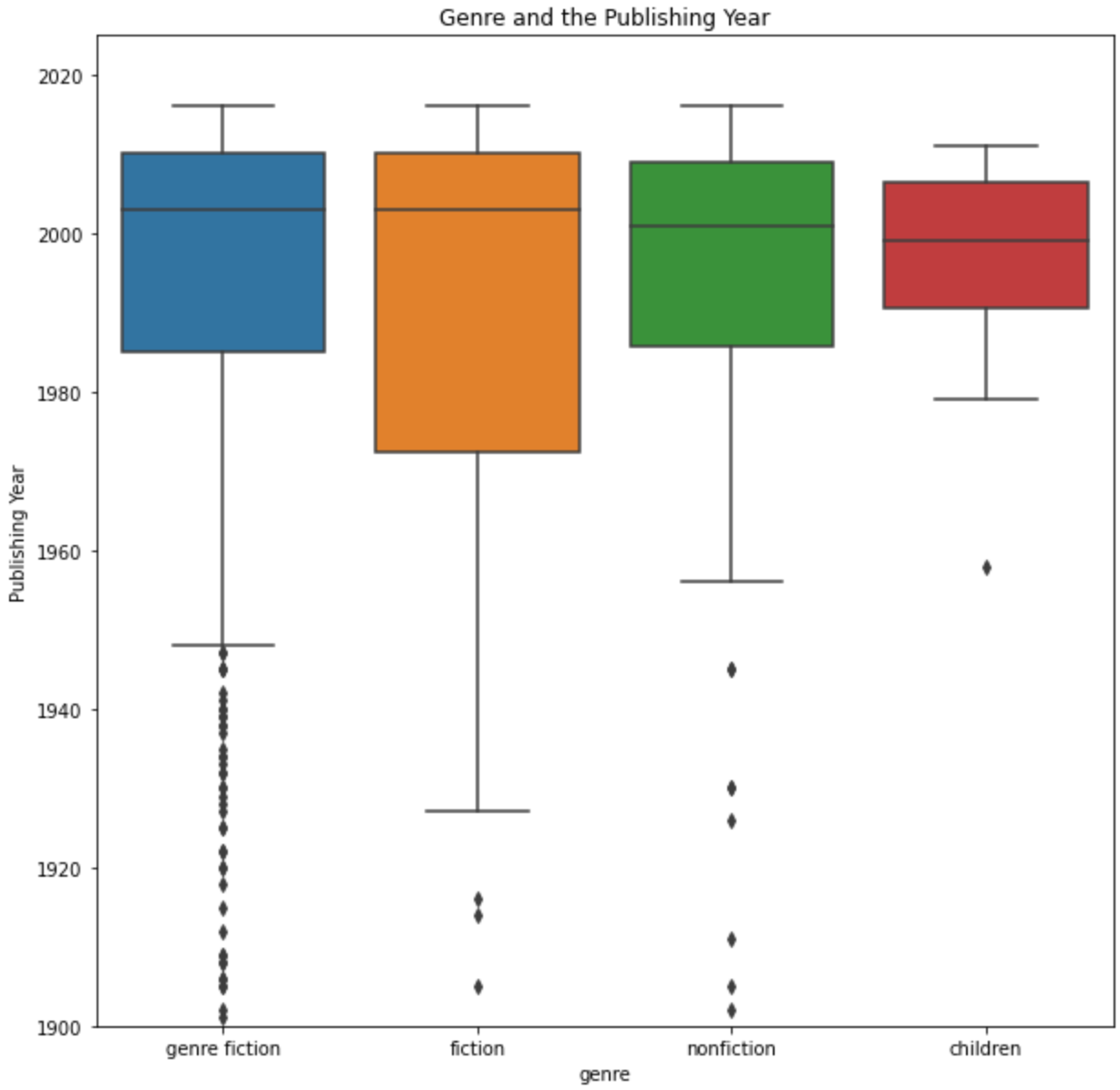


The scatter plot above shows that there were more number of books sold after the Year 2000. The highest number of books sold seems to lie between the years 2000 and 2020. The Publishing year has a small significant impact on the number of books being sold.

In [42]:

```
#Now, we will look at the relationship between the Publishing Year and genre
# Plot the box plot
plt.figure(figsize=(10,10))
Popular_genre_in_year = sns.boxplot(data = new_book_dataset_df, x='genre', y='Publishing Y
Popular_genre_in_year.set_ylim(1900,2025)
# The title and the x and y-axis labels are added
plt.xlabel('genre')
plt.ylabel('Publishing Year')
plt.title('Genre and the Publishing Year')

# The box plot is displayed
plt.show()
```

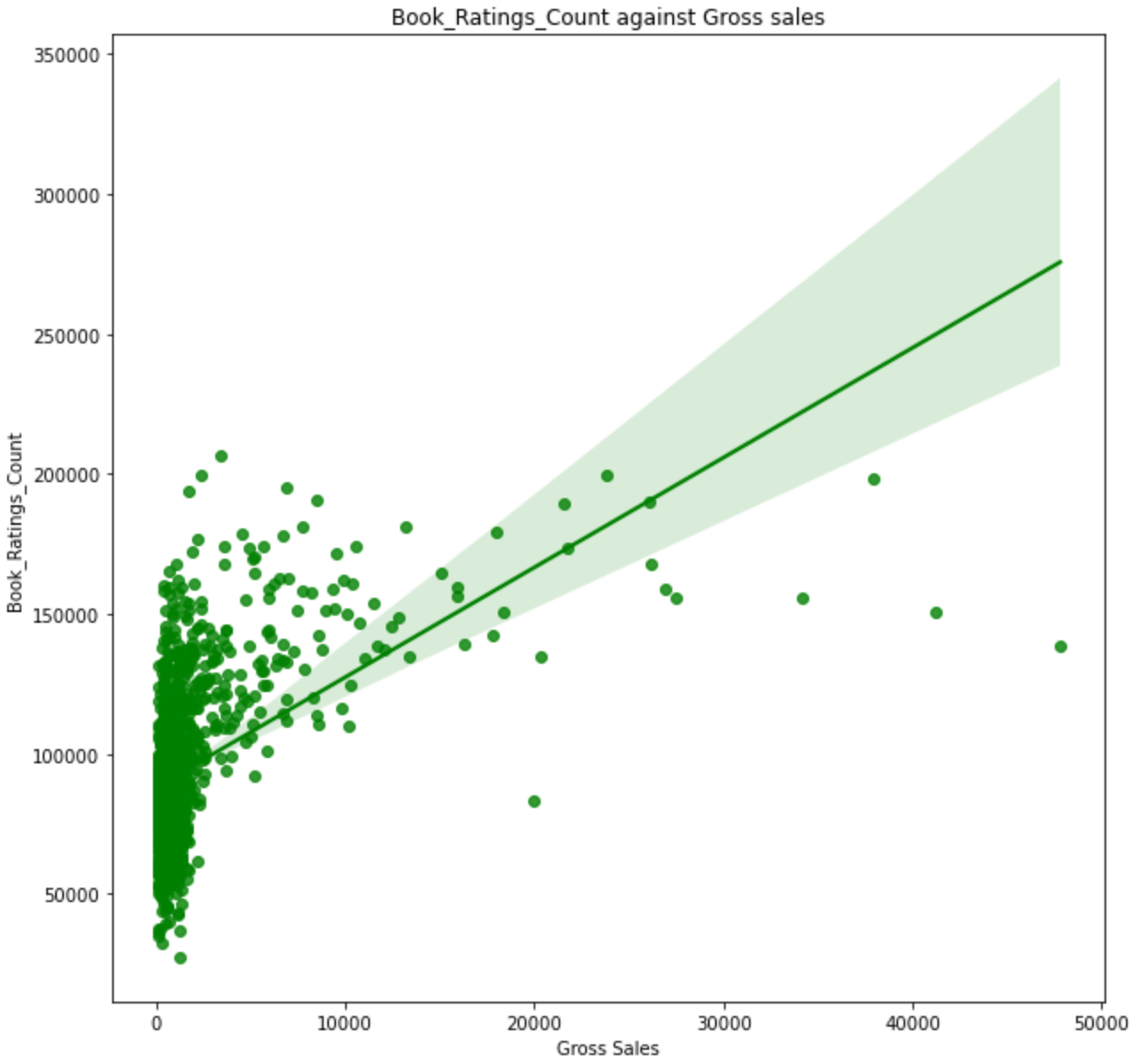


The genre titled Fiction seems to be popular from 1970 to 2010. It seems to be the only genre that has been popular among readers for the most amount of time. The genre of non-fiction only seems to be popular from 1980 till the early 2000s. There seem to be outliers present in the data points for the gesture genre that do not fall under the range of books published in the quartiles of the Publishing Year.

In [43]:

```
#How does the Book_rating_count affects gross sales
#Plot the graph
plt.figure(figsize=(10,10))
Book_rating_and_gross_sales = sns.regplot(data = new_book_dataset_df, x='gross sales', y='
# The title and the x and y-axis labels are added
plt.xlabel('Gross Sales')
plt.ylabel('Book_Ratings_Count')
plt.title('Book_Ratings_Count against Gross sales')
#Display the graph
print(Book_rating_and_gross_sales)
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

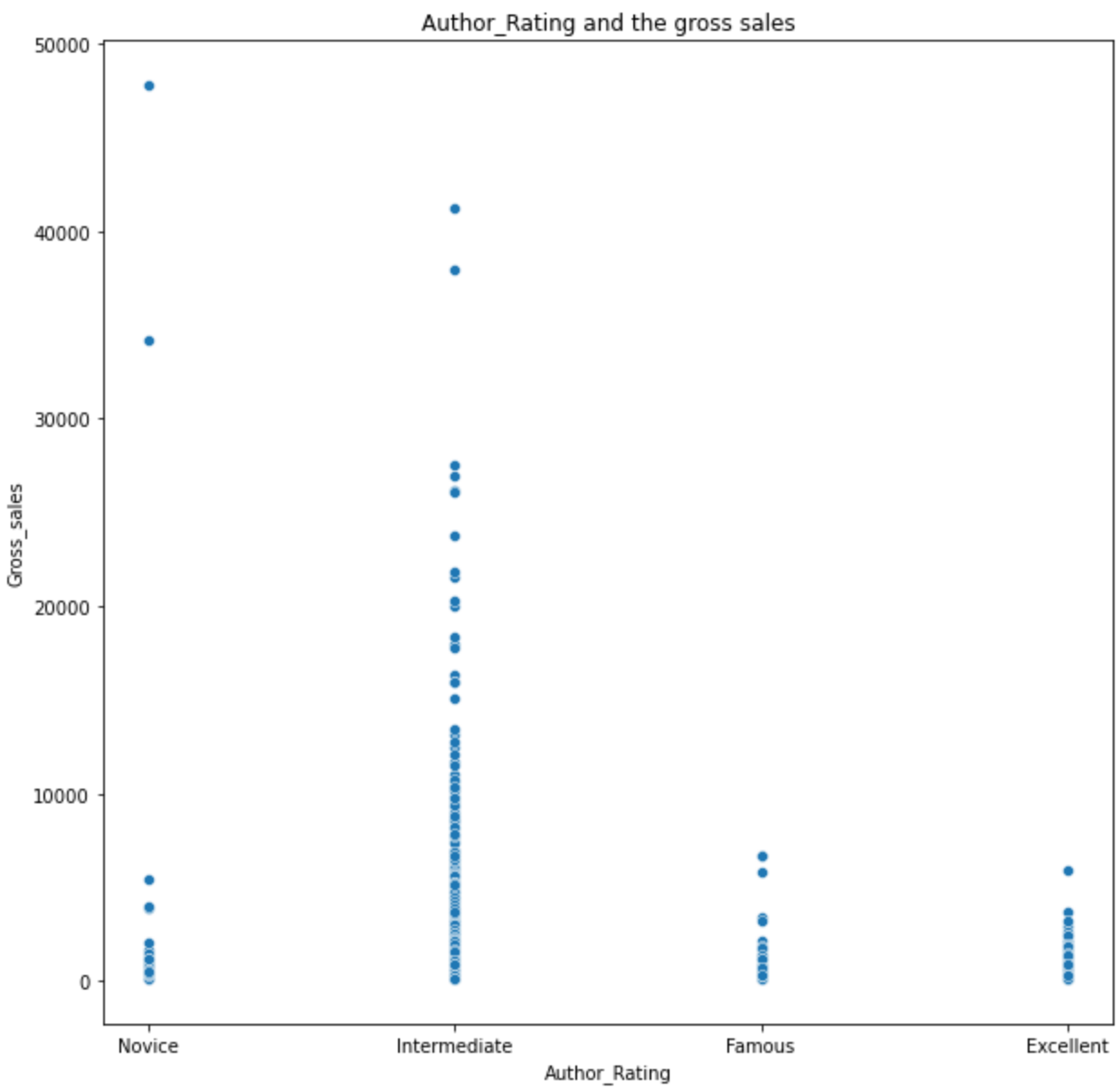


The higher the gross sales, the lower the book ratings count. This can be interpreted from the regression plot above as the number of gross sales increases, the book ratings count should increase but the books with a lower than predicted ratings count tend to yield more revenue. Hence, the book ratings count does not have a significant impact on the gross sales.

In [43]:

```
#How does the Author_Rating affects gross sales
# Plot the scatter plot
plt.figure(figsize=(10,10))
Author_Rating_gross_sales = sns.scatterplot(data = new_book_dataset_df, x='Author_Rating',
# The title and the x and y-axis labels are added
plt.xlabel('Author_Rating')
plt.ylabel('Gross_sales')
plt.title('Author_Rating and the gross sales')

# The box plot is displayed
plt.show()
```

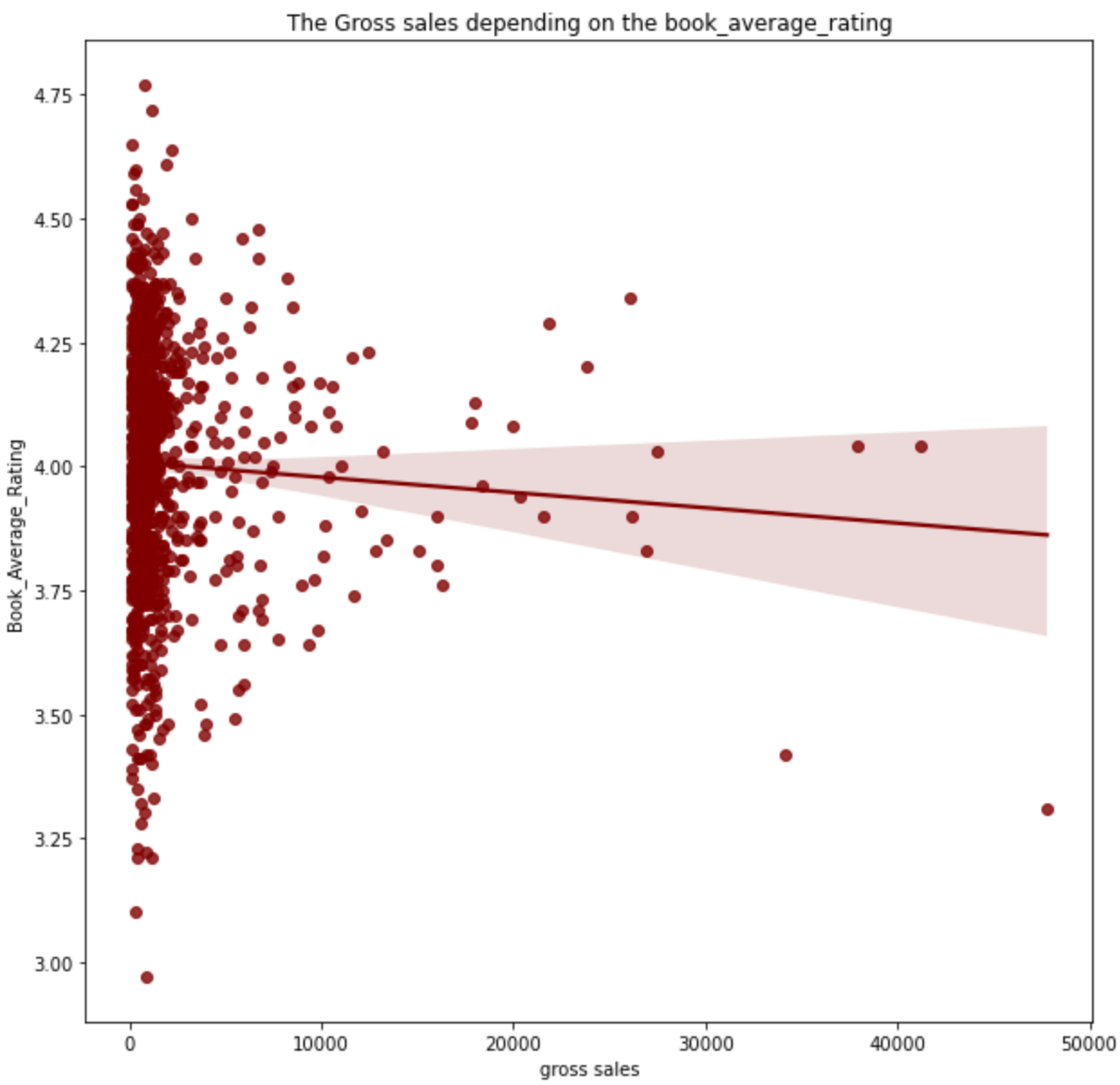


The higher the Author rating, the lower the gross sales. This can be seen from the scatter plot above which showcases that the Author_Rating of Intermediate and Novice tend to yield the highest sales together. This shows that the Author_Rating does not affect the gross sales.

In [44]:

```
#How does the Book_average_rating affects gross sales
#Plot the graph
plt.figure(figsize=(10,10))
Book_average_rating_and_gross_sales = sns.regplot(data = new_book_dataset_df, x='gross sal
# The title and the x and y-axis labels are added
plt.xlabel('gross sales')
plt.ylabel('Book_Average_Rating')
plt.title('The Gross sales depending on the book_average_rating')
#Display the graph
print(Book_average_rating_and_gross_sales)
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

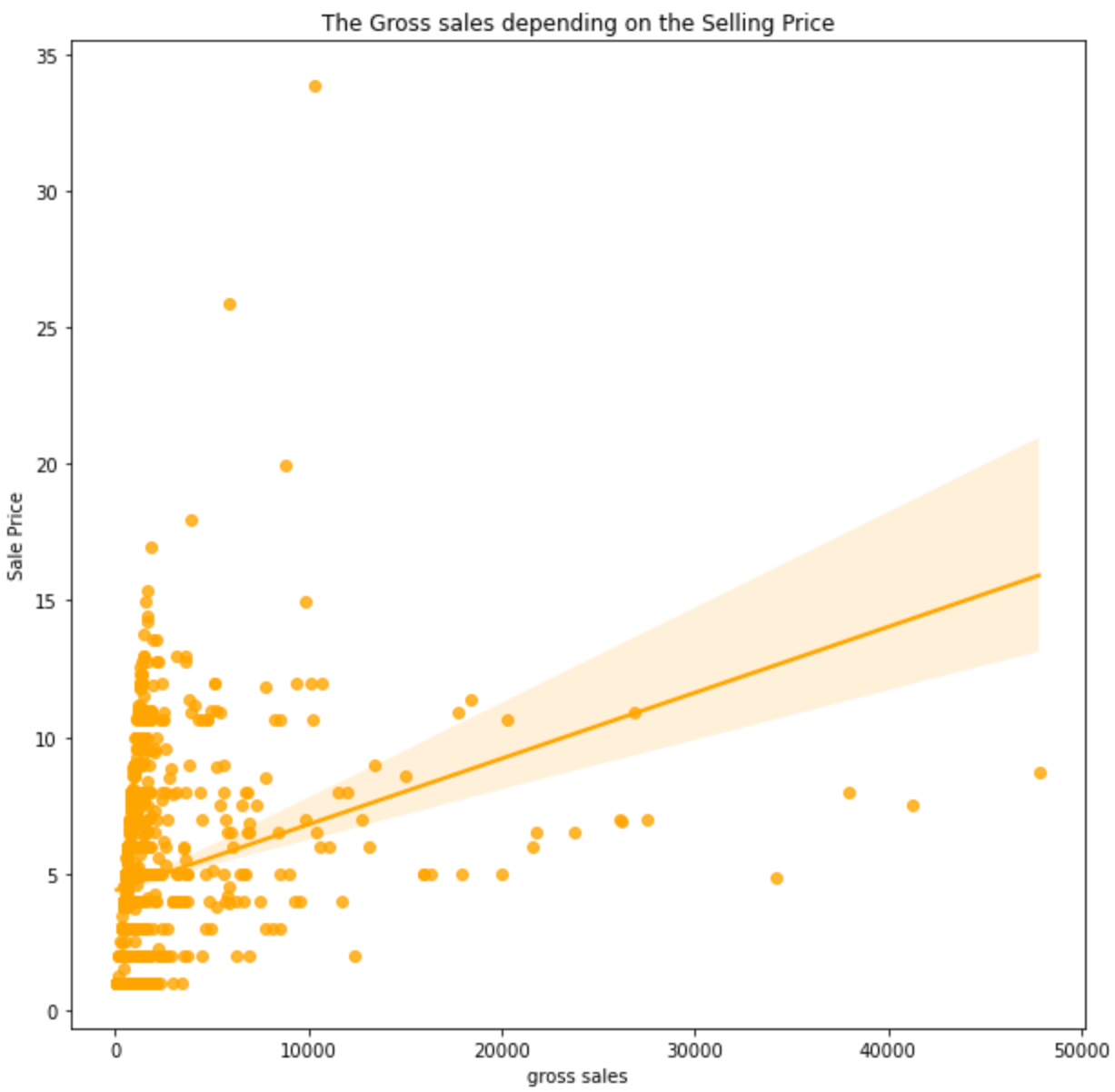


As seen from the graph above, books with a higher average rating tend to yield more gross sales compared to books with a lower rating. The graph also showcases that books with a higher rating tend to be sold more than what is expected. Hence, this proves that the average rating of the book does affect the gross sales as readers tend to purchase books with higher ratings.

In [45]:

```
#How does the Sales price affects gross sales
#Plot the regression plot
plt.figure(figsize=(10,10))
Sale_price_and_gross_sales = sns.regplot(data = new_book_dataset_df, x='gross sales', y='s
# The title and the x and y-axis labels are added
plt.xlabel('gross sales')
plt.ylabel('Sale Price')
plt.title('The Gross sales depending on the Selling Price')
#Display the regression plot
print(Sale_price_and_gross_sales)
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

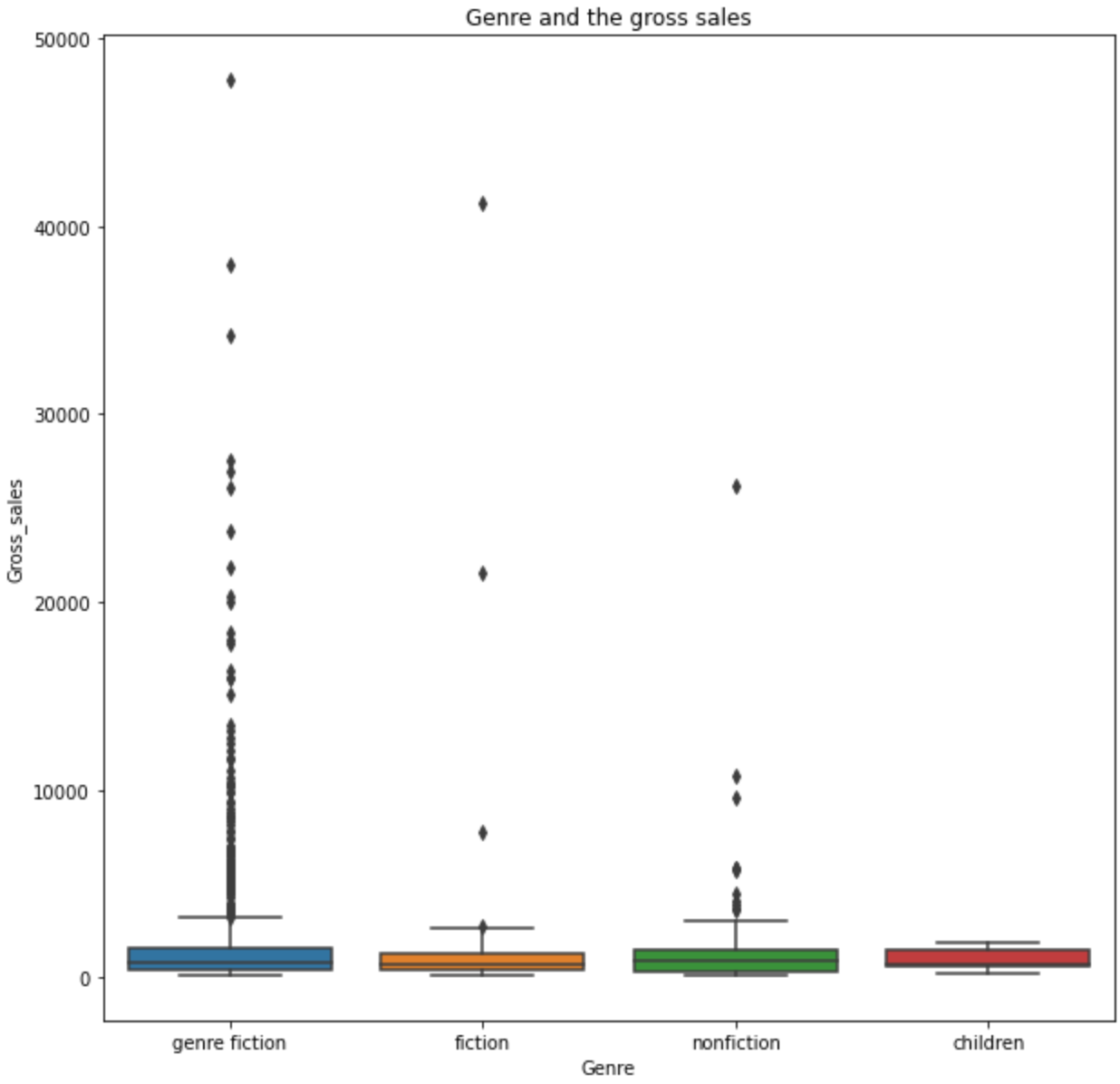


As seen from the graph above, books with a lower sale price tend to yield more gross sales compared to books with a higher sale price. The graph also showcases that books with a lower sale price tend to be sold more as they are more affordable and yield a larger revenue as more readers tend to purchase them. This shows that the sale price does affect the gross sales as readers tend to prefer books with low sale prices.

In [44]:

```
#How does the genre affects gross sales
# Plot the box plot
plt.figure(figsize=(10,10))
Genre_gross_sales = sns.boxplot(data = new_book_dataset_df, x='genre', y='gross sales')
# The title and the x and y-axis labels are added
plt.xlabel('Genre')
plt.ylabel('Gross_sales')
plt.title('Genre and the gross sales')

# The box plot is displayed
plt.show()
```



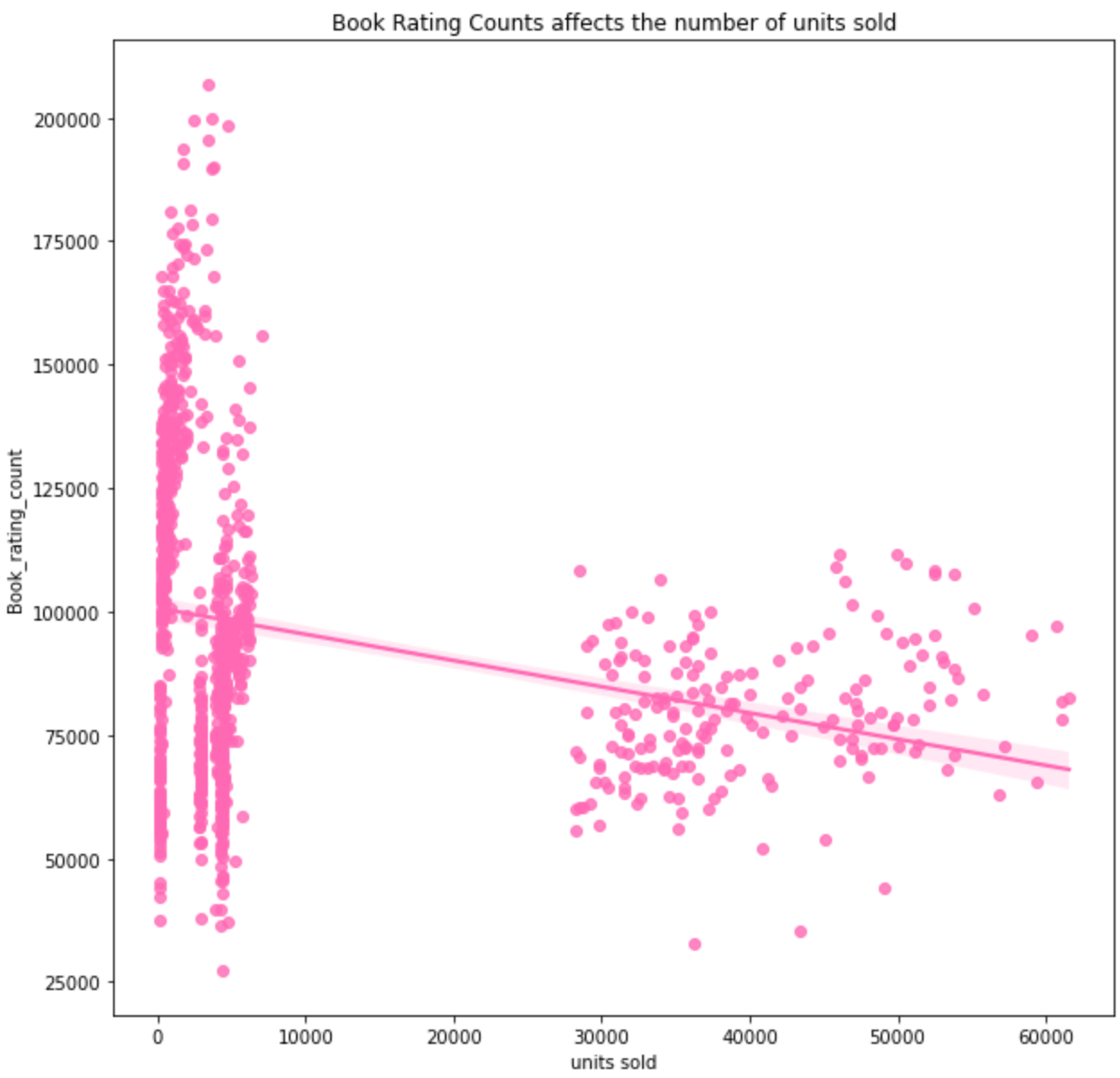
As seen from the boxplot, the genre does not seem to affect the gross sales of the books as most of the books from all the genres seem to yield a lower gross sale and do not affect the revenue generated from the type of book that is being sold.

In [47]:

```
#How does the rating of the book affect the number of units of books sold
#Plot the graph
plt.figure(figsize=(10,10))
Book_rating_and_units_sold = sns.regplot(data = new_book_dataset_df, x='units sold', y='Bo
# The title and the x and y-axis labels are added
plt.xlabel('units sold')
plt.ylabel('Book_rating_count')
plt.title('Book Rating Counts affects the number of units sold')

#Display the graph
print(Book_rating_and_units_sold)
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

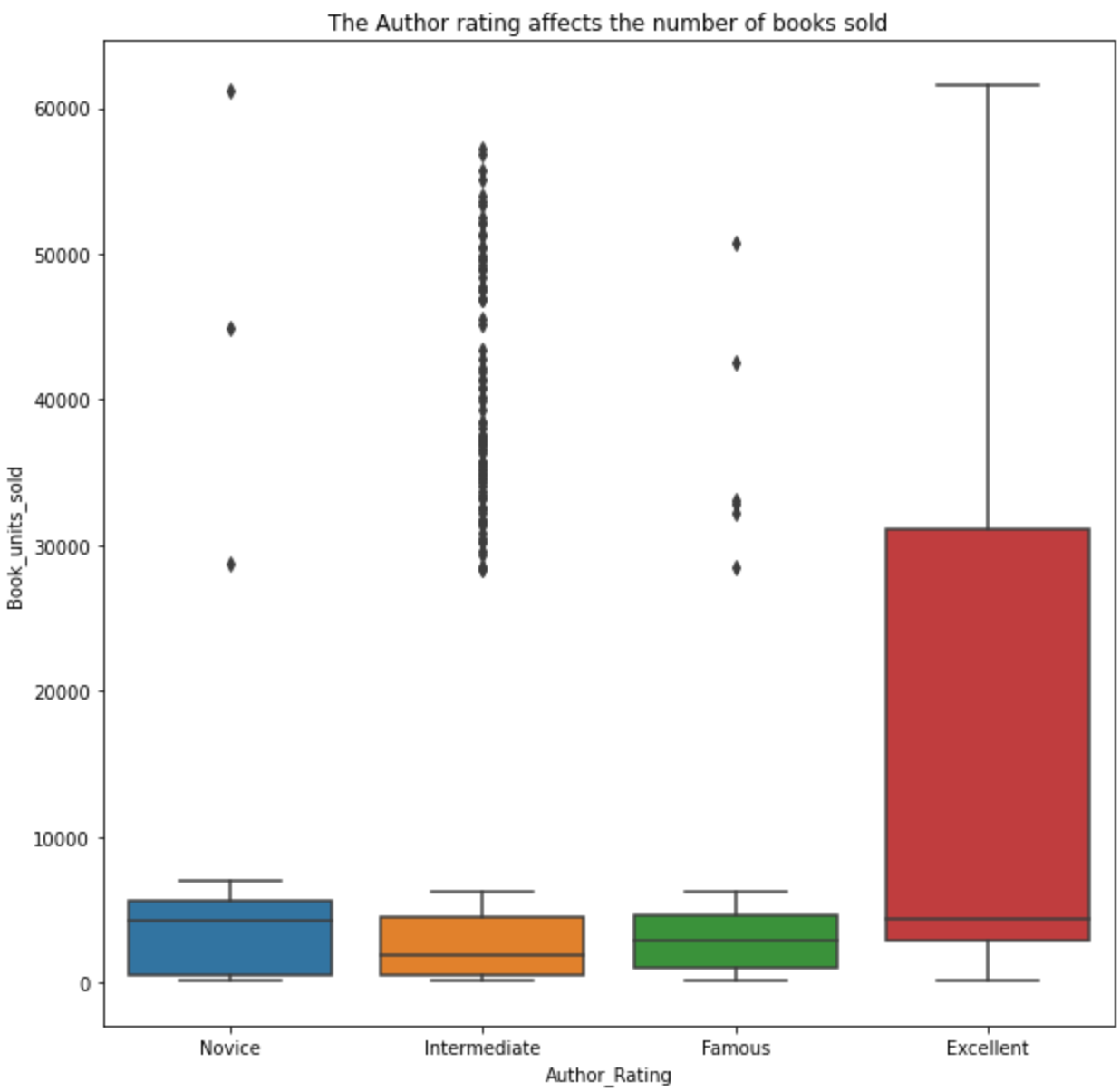


As seen from the graph above, as the number of books sold increases, the book rating count decreases constantly. The graph also showcases that books with a higher rating tend to be sold less than books with ratings below the average. Hence, this proves that the rating of the book does not affect the number of books being sold as readers still tend to purchase books with lower ratings.

In [48]:

```
#How does the Author_Rating affect the number of units of books sold
#Plot the boxplot
plt.figure(figsize=(10,10))
Author_rating_units_sold = sns.boxplot(data = new_book_dataset_df, x='Author_Rating', y='u
# The title and the x and y-axis labels are added
plt.xlabel('Author_Rating')
plt.ylabel('Book_units_sold')
plt.title('The Author rating affects the number of books sold')
#Display the boxplot
print(Author_rating_units_sold)
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

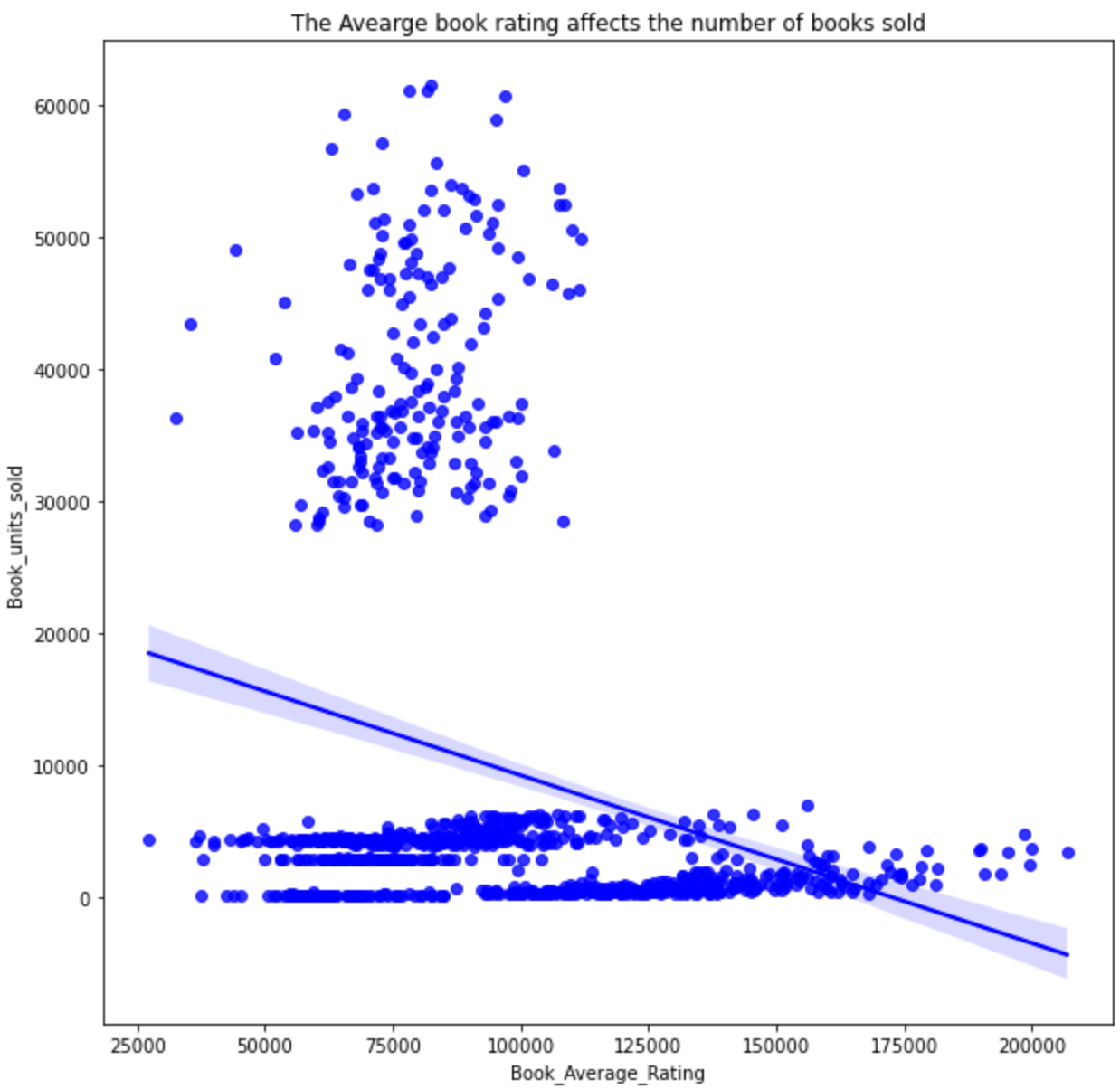


The higher the Author_Rating, the higher the number of books sold. This can be interpreted from the boxplot above which shows that books with an Author_Rating of Excellent have been sold the most compared to the other three ratings. Hence, the Author_Rating affects the number of books sold and should be taken into consideration when building our model.

In [49]:

```
#How does the Book_average_rating affect the number of units of books sold
#Plot the boxplot
plt.figure(figsize=(10,10))
Book_Average_rating_units_sold = sns.regplot(data = new_book_dataset_df, x='Book_ratings_c
# The title and the x and y-axis labels are added
plt.xlabel('Book_Average_Rating')
plt.ylabel('Book_units_sold')
plt.title('The Aveargе book rating affects the number of books sold')
#Display the boxplot
print(Book_Average_rating_units_sold)
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

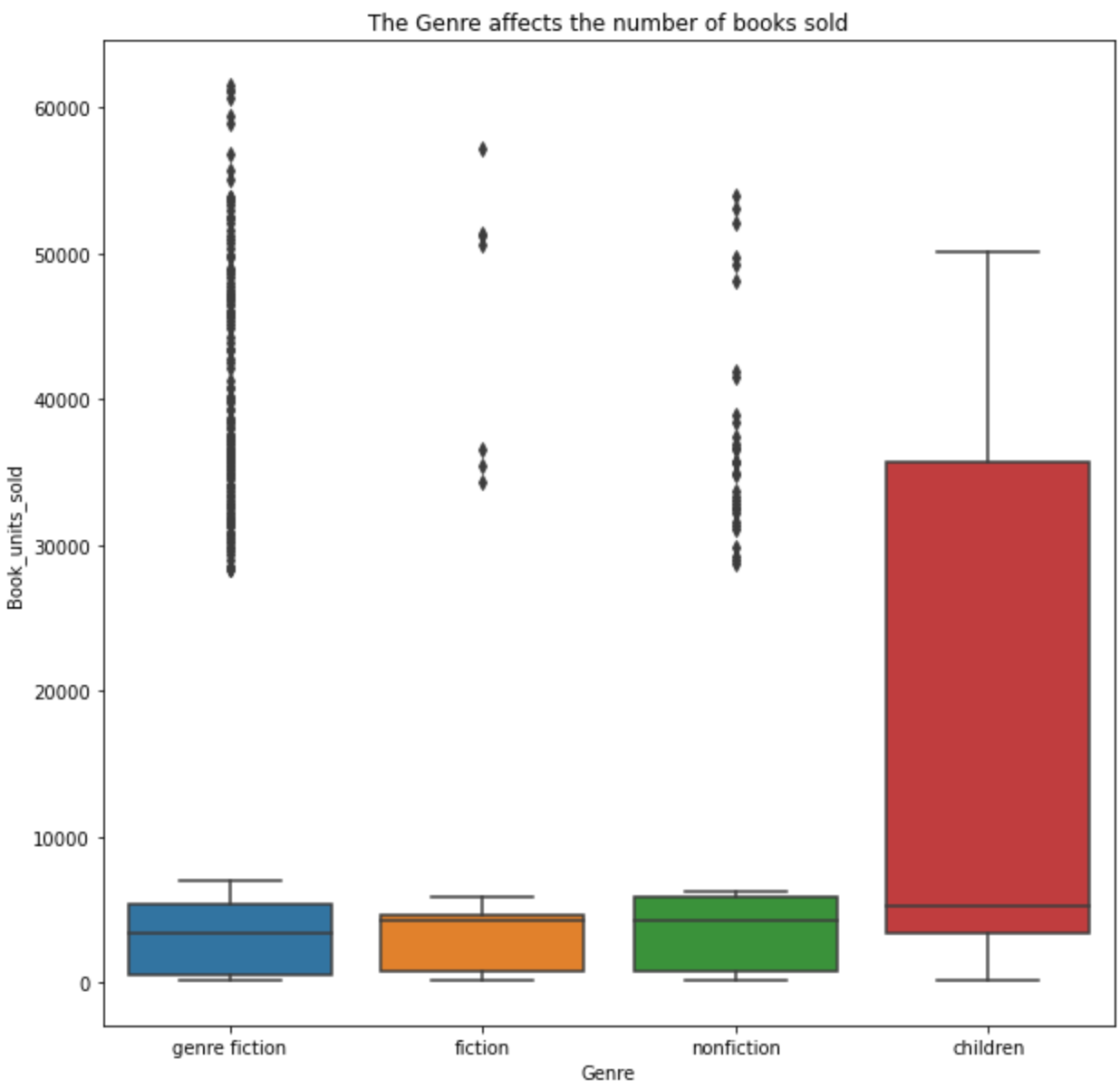


The lower the average book rating, the higher the number of books sold. This can be interpreted from the graph above which displays that more books are sold when the average book rating is lower. Hence, we can conclude that the average book rating has no impact on the number of books sold as readers tend to consider other factors when purchasing a book.

In [50]:

```
#How does the genre affect the number of units of books sold
#Plot the boxplot
plt.figure(figsize=(10,10))
Genre_units_sold = sns.boxplot(data = new_book_dataset_df, x='genre', y='units sold')
# The title and the x and y-axis labels are added
plt.xlabel('Genre')
plt.ylabel('Book_units_sold')
plt.title('The Genre affects the number of books sold')
#Display the boxplot
print(Book_Average_rating_units_sold)
```

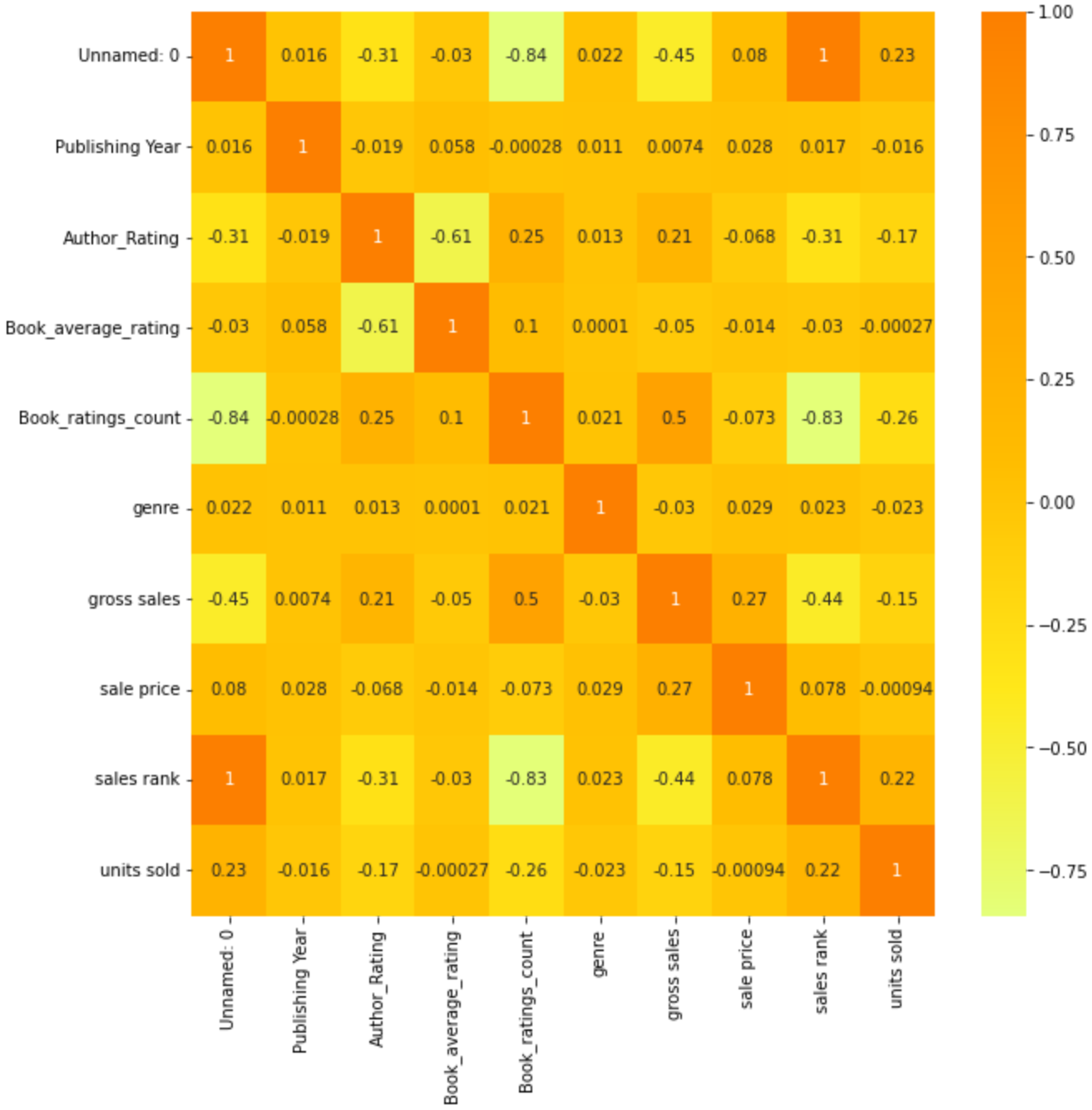
```
AxesSubplot(0.125,0.125;0.775x0.755)
```



The genre affects the number of units of books sold. Despite the genre of children having wide variability, the highest units of books sold are from the genre fiction and that tends to fall out of the boxplot as outliers. Hence, the feature genre does affect the number of books sold and should be taken into consideration when building our model.

In [318…

```
#To view how closely related all the features are
plt.figure(figsize=(10,10))
#Load the heatmap to view the correlation between features
Book_correlation_matrix = sns.heatmap(new_book_dataset_df.corr(), cmap="Wistia", annot = T
#Display the heatmap
plt.show()
```



The Heat Map above displays the correlation between features that have numeric datatypes and the features Author_Rating and genre are not included in this heat map as there are features with non-numeric values. The features sale rank and sale price tend to have the closest relation with the number of books sold and hence will be used to train the machine learning model.

Which visualisation (of the ones you prepared) is most important and why?

The regression plots for the number of books sold and the heat map are the most important visualizations that are needed to view the relationship between the other features and the number of books sold. However, the most effective of them all is the heatmap as it displays the correlation between the features and the units of books sold, and the features with the closest relation to the units of books sold can be identified and used to train the machine learning model.

6. Build your ML (Machine Learning) model

The Aim: To build a linear regression model that predicts the amount that should be kept in stock to generate revenue.

The Features chosen: Sales rank and sale price to predict the number of units of books to be sold

In [301…
```
#Reduce the dataset to only 100 values
new_book_dataset_df_100 = new_book_dataset_df[:][:100]
#Convert the features into arrays
model_X = np.array(new_book_dataset_df_100['sales rank'], new_book_dataset_df_100['sale pr
model_Y = np.array(new_book_dataset_df_100['units sold']).reshape(-1,1)
```

In [302…
```
#Data is split into training and testing datasets
model_X_train, model_X_test, model_Y_train, model_Y_test = train_test_split(model_X,model_
```

In [303…
```
#Model is created and being fitted
Linear_reg_model = LinearRegression()
Linear_reg_model.fit(model_X_train, model_Y_train)
```

Out[303…
```
LinearRegression()
```

In [304…
```
#The score of the model
print(Linear_reg_model.score(model_X_test,model_Y_test))
```

```
0.8700448036576356
```

In [305…
```
#The model prediction for the units of books to be stored
Predict_Y = Linear_reg_model.predict(model_X_test)
print(Predict_Y)
```

```
[[1677.57909756]
 [2226.21876416]
 [1986.18891002]
 [2054.76886835]
 [3563.52795149]
 [1368.9692851 ]
 [ 888.90957683]
 [1437.54924343]
 [3563.52795149]
 [ 957.48953516]
 [1643.2891184 ]
 [3734.9778473 ]
 [ 648.8797227 ]
 [1711.86907673]
 [1883.31897254]
 [2946.30832657]
 [ 305.97993107]
 [1197.51938929]
 [3289.20811819]
 [2157.63880583]
 [1540.41918091]
 [2191.928785  ]
 [1197.51938929]
 [ 340.26991023]
 [3494.94799316]]
```

In [ ]:

7. Validation

In [306…
```
#Validate using Cross_validation
Cross_val_folds = KFold(n_splits = 5, shuffle = True, random_state=250)
val_score = cross_val_score(Linear_reg_model, model_X_train, model_Y_train, scoring='r2',
print(val_score)
```

```
[ 0.80615922  0.17450334  0.67810608 -0.71357095  0.64011961]
```

The Cross-Validation score above shows that the model can predict the units of books sold quite accurately as the validation scores for most of the folds are high and hence the model has high precision. However, we still need to calculate the Root mean square error score and the mean square error score to determine how error-free the model is.

In [309…
```
#The Root mean square error score
R2_score = np.sqrt(metrics.mean_squared_error(model_Y_test, Predict_Y))
print(R2_score)
```

```
318.2928149021665
```

In [437…
```
#Mean Square error
Book_mean_square_error = np.square(np.subtract(model_Y_test, Predict_Y)).mean()
print(Book_mean_square_error)
```

```
3402450.9683409496
```

8. Feature engineering

In [313…
```
#We can use the genre and Author_Rating to improve the accuracy of the model
#Label Encoding for both features to convert the strings into integers for these two categ
#Import the library
from sklearn import preprocessing
#Load the encoder
Label_Encoder_Author_Rating = preprocessing.LabelEncoder()
#Fit the Author_Rating Feature into the Label Encoder
new_book_dataset_df['Author_Rating'] = Label_Encoder_Author_Rating.fit_transform(new_book_
#Display the values after encoding
print(new_book_dataset_df['Author_Rating'])
```

```
0      3
1      2
2      3
3      2
4      2
      ..
993    2
994    0
995    2
996    0
997    0
Name: Author_Rating, Length: 998, dtype: int32
```

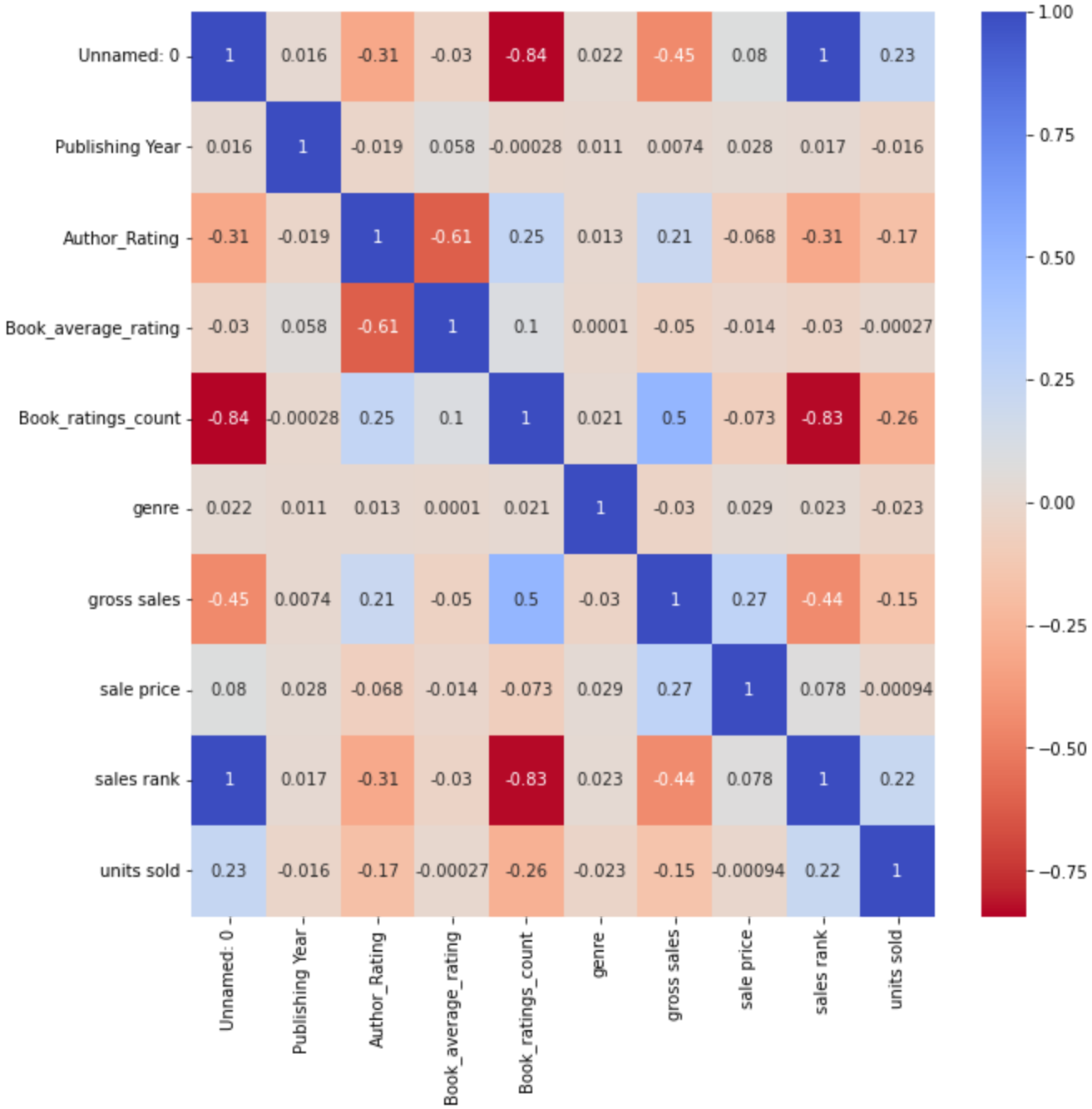
In [314…
```
#Now, let's repeat the same steps for genre
#Load the encoder
Label_Encoder_Author_Rating = preprocessing.LabelEncoder()
#Fit the genre Feature into the Label Encoder
new_book_dataset_df['genre'] = Label_Encoder_Author_Rating.fit_transform(new_book_dataset_
#Display the values after encoding
print(new_book_dataset_df['genre'])
```

```
0      2
1      2
```

```
2       2
3       1
4       2
       ..
993     3
994     2
995     2
996     2
997     2
Name: genre, Length: 998, dtype: int32
```

In [435…

```
#Now, let us view the correlation matrix between these two new features along with the res
plt.figure(figsize=(10,10))
#Load the heatmap to view the correlation between features
New_Book_correlation_matrix = sns.heatmap(new_book_dataset_df.corr(), cmap="coolwarm_r", a
#Display the heatmap
plt.show()
```



In [ ]:

In [427…
```
#Rebuild the model
#Reduce the dataset to only 100 values
new_book_dataset_df_100 = new_book_dataset_df[:][:100]
#Convert the features into arrays
model_X_1 = np.array(new_book_dataset_df_100['genre'], new_book_dataset_df_100['Author_Rat
model_Y_1 = np.array(new_book_dataset_df_100['units sold']).reshape(-1,1)
```

In [428…
```
#Data is split into training and testing datasets
model_X_1_train, model_X_1_test, model_Y_1_train, model_Y_1_test = train_test_split(model_
```

In [432…
```
#Model is created and being fitted
Linear_reg_model = LinearRegression()
Linear_reg_model.fit(model_X_1_train, model_Y_1_train)
```

Out[432…
```
LinearRegression()
```

In [433…
```
#The score of the model
print(Linear_reg_model.score(model_X_1_test,model_Y_1_test))
```

```
0.02771845596147171
```

In [434…
```
#Validate using Cross_validation
Cross_val_folds = KFold(n_splits = 5, shuffle = True, random_state=250)
val_score = cross_val_score(Linear_reg_model, model_X_1_train, model_Y_1_train, scoring='r
print(val_score)
```

```
[-0.0292551  -0.16863972 -0.4552006  -0.00387924 -0.11058745]
```

In [ ]:
```
```

10. Evaluation of your model (200 to 400 words)

The linear regression model that I have built using the book dataset has an accuracy score of 0.87, which shows that the model has higher accuracy in predicting the number of units of books that have to be kept in stock and sold to achieve higher gross sales. Cross-validation is also performed on the linear regression model which yields a high coefficient value for fairly every fold which showcases that the model has a high performance rate. The mean squared error is also calculated to view how closely the data points are spread to the regression line. The high mean squared value of 3402450 shows that the data points for the features' sale price and sale rank are spread widely around the mean and that the model has higher errors and the predictions are less precise. On the other hand, the Root Mean Square Error(RMSE) which calculates the standard deviation for the predicted errors shows a smaller value of 318. This means that there are more data points of the two features mentioned above concentrated along the best-fit line. However, there are still errors present in the model and its prediction. The model is set to contain errors due to the low correlation matrix value between the features' sale price and sale rank as seen from the heat map. The heat map only showcases correlations between features that have numeric values.

The features of Author Rating and Genre are not taken into consideration since they contain non-numeric datatypes. Hence, I have decided to perform Feature Extraction on these two features. I made use of the Label Encoding technique where the datatype of the values of two features will be converted to integers. The disadvantage of using this method is that the order of the values will be disoriented where the value with a higher number is considered to be more important than that of a lower value. For example, when the genre

feature is converted using the Label Encoder it assigns an integer value of 0,2,3 to the genres Fiction, Non-fiction, and Children. Now, the genre Children seems to have higher importance than the genre Fiction when in reality, Fiction is the most popular genre. This eventually affects the correlation between the number of units of books sold and the features of Author Rating and Genre. When the model is retrained using these two features, it produces a low accuracy score and is hence deemed to have a lower prediction rate as seen from the low validation score.

In Conclusion, the linear regression model is only accurate to a certain extent in predicting the number of books to be kept in stock to generate more revenue. The model has to be trained further and maybe a new dataset might be used to retrain the model as the current book dataset seems to contain a significant amount of errors and does not seem to have a high correlation between its features. The model might work better on other datasets.

References

1. Murrey, J. (2024) Books sales and ratings, Kaggle. Available at: https://www.kaggle.com/datasets/thedevastator/books-sales-and-ratings (Accessed: 05 January 2024).
2. Rahulsingh, D. (2023) Best-selling-books, Kaggle. Available at: https://www.kaggle.com/datasets/drahulsingh/best-selling-books (Accessed: 05 January 2024).
3. Milliot, J. (2023) Nea finds worrying drop in reading participation, PublishersWeekly.com. Available at: https://www.publishersweekly.com/pw/by-topic/industry-news/bookselling/article/93659-nea-finds-worrying-drop-in-reading-participation.html (Accessed: 05 January 2024).
4. Joshua, S. (2022) How to combine multiple CSV files using Python for your analysis, Medium. Available at: https://medium.com/@stella96joshua/how-to-combine-multiple-csv-files-using-python-for-your-analysis-a88017c6ff9e (Accessed: 05 January 2024).
5. Pandas.merge (2023) pandas.merge - pandas 2.1.4 documentation. Available at: https://pandas.pydata.org/docs/reference/api/pandas.merge.html (Accessed: 05 January 2024).
6. Young, S. (2023) Are we losing the ability to read books?, Scott H Young. Available at: https://www.scotthyoung.com/blog/2023/03/21/reading-fewer-books/ (Accessed: 05 January 2024).
7. How to drop one or multiple columns in pandas dataframe (2023) GeeksforGeeks. Available at: https://www.geeksforgeeks.org/how-to-drop-one-or-multiple-columns-in-pandas-dataframe/ (Accessed: 05 January 2024).
8. Pandas.merge (2023) pandas.merge - pandas 2.1.4 documentation. Available at: https://pandas.pydata.org/docs/reference/api/pandas.merge.html (Accessed: 05 January 2024).
9. user20320394user20320394 and TheMasterTheMaster 46.8k66 gold badges6464 silver badges8989 bronze badges (2023) Convert data to 1NF, Stack Overflow. Available at: https://stackoverflow.com/questions/74179423/convert-data-to-1nf (Accessed: 05 January 2024).
10. Mussa, B. (2022) Calculating mathematical range in python for pandas describe, Medium. Available at: https://bilalmussa.medium.com/calculating-mathematical-range-in-python-for-pandas-describe-50898bae7405 (Accessed: 05 January 2024).

11. Pandas DataFrame std() Method (no date) Pandas DataFrame STD() method. Available at: https://www.w3schools.com/python/pandas/ref_df_std.asp#:~:text=The%20var()%20method%20calculates,the% (Accessed: 05 January 2024).

12. Zach (2023) How to calculate quartiles in Pandas (with example), Statology. Available at: https://www.statology.org/pandas-quartiles/ (Accessed: 05 January 2024).

13. Ghosh, T.K. (2023) How to set axes labels & limits in a seaborn plot?, Tutorialspoint. Available at: https://www.tutorialspoint.com/how-to-set-axes-labels-amp-limits-in-a-seaborn-plot (Accessed: 05 January 2024).

14. Stojikovic, M. (2023) Linear regression in python, Real Python. Available at: https://realpython.com/linear-regression-in-python/#multiple-linear-regression (Accessed: 05 January 2024).

15. Biswal, A. (2023) Sklearn linear regression (step-by-step explanation): Sklearn Tutorial, Simplilearn.com. Available at: https://www.simplilearn.com/tutorials/scikit-learn-tutorial/sklearn-linear-regression-with-examples (Accessed: 05 January 2024).

16. Mali, K. (2023) Everything you need to know about linear regression!, Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/ (Accessed: 05 January 2024).

17. Jnikhilsai (2019) Cross-validation with linear regression, Kaggle. Available at: https://www.kaggle.com/code/jnikhilsai/cross-validation-with-linear-regression (Accessed: 05 January 2024).

18. BOZKUŞ, E. (2022) Multiple linear regression in python, Kaggle. Available at: https://www.kaggle.com/code/emineyetm/multiple-linear-regression-in-python (Accessed: 05 January 2024).

19. Gupta, A. (2023) Mean squared error: Overview, examples, concepts and more: Simplilearn, Simplilearn.com. Available at: https://www.simplilearn.com/tutorials/statistics-tutorial/mean-squared-error (Accessed: 05 January 2024).

20. BengFort, B. (2021) Cross validation scores, Cross Validation Scores - Yellowbrick v1.5 documentation. Available at: https://www.scikityb.org/en/latest/api/model_selection/cross_validation.html#:~:text=Generally%20we%20deter (Accessed: 05 January 2024).

21. RMSE: Root mean square error (2023) Statistics How To. Available at: https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/ (Accessed: 05 January 2024).

In [ ]: