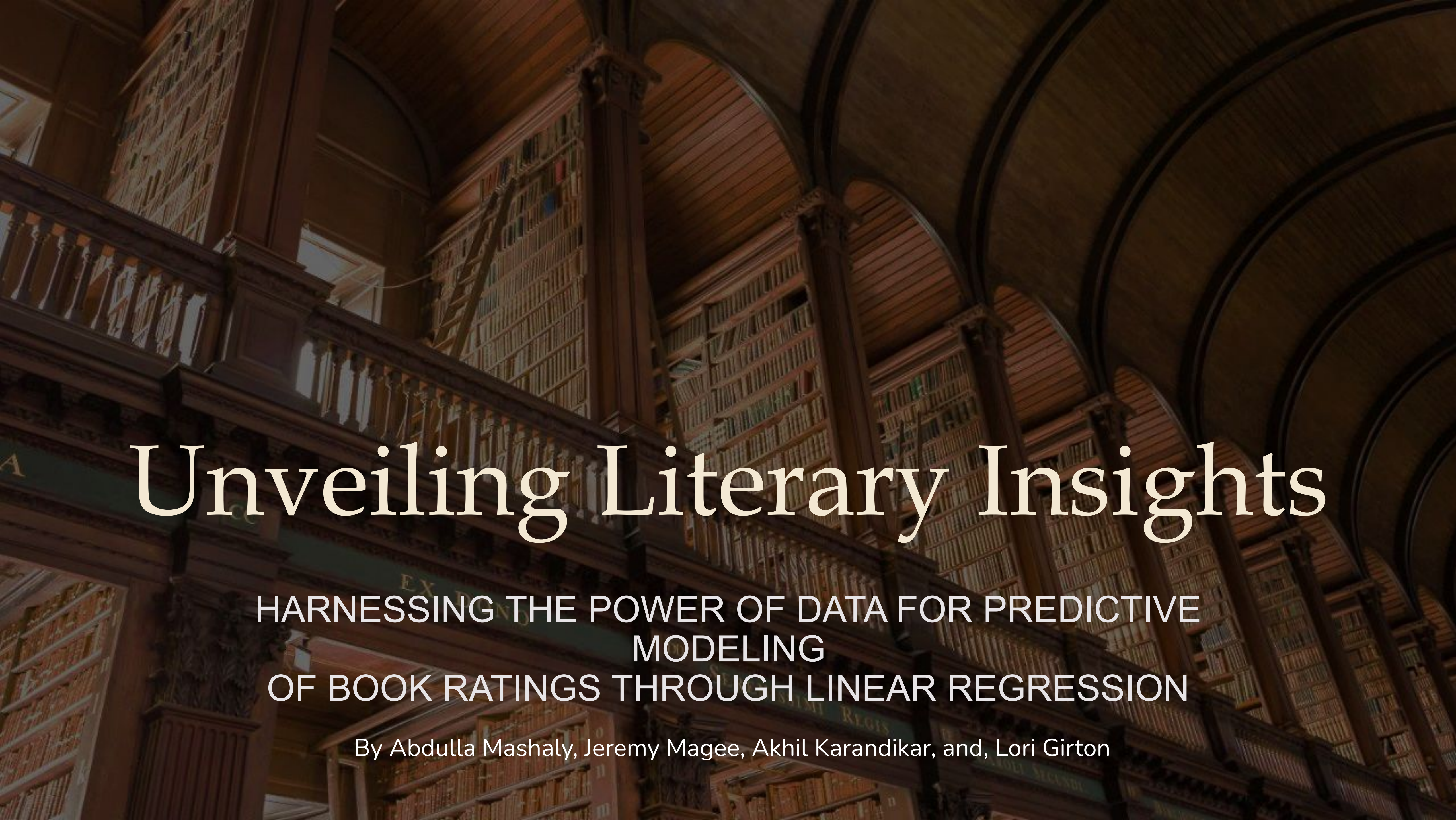


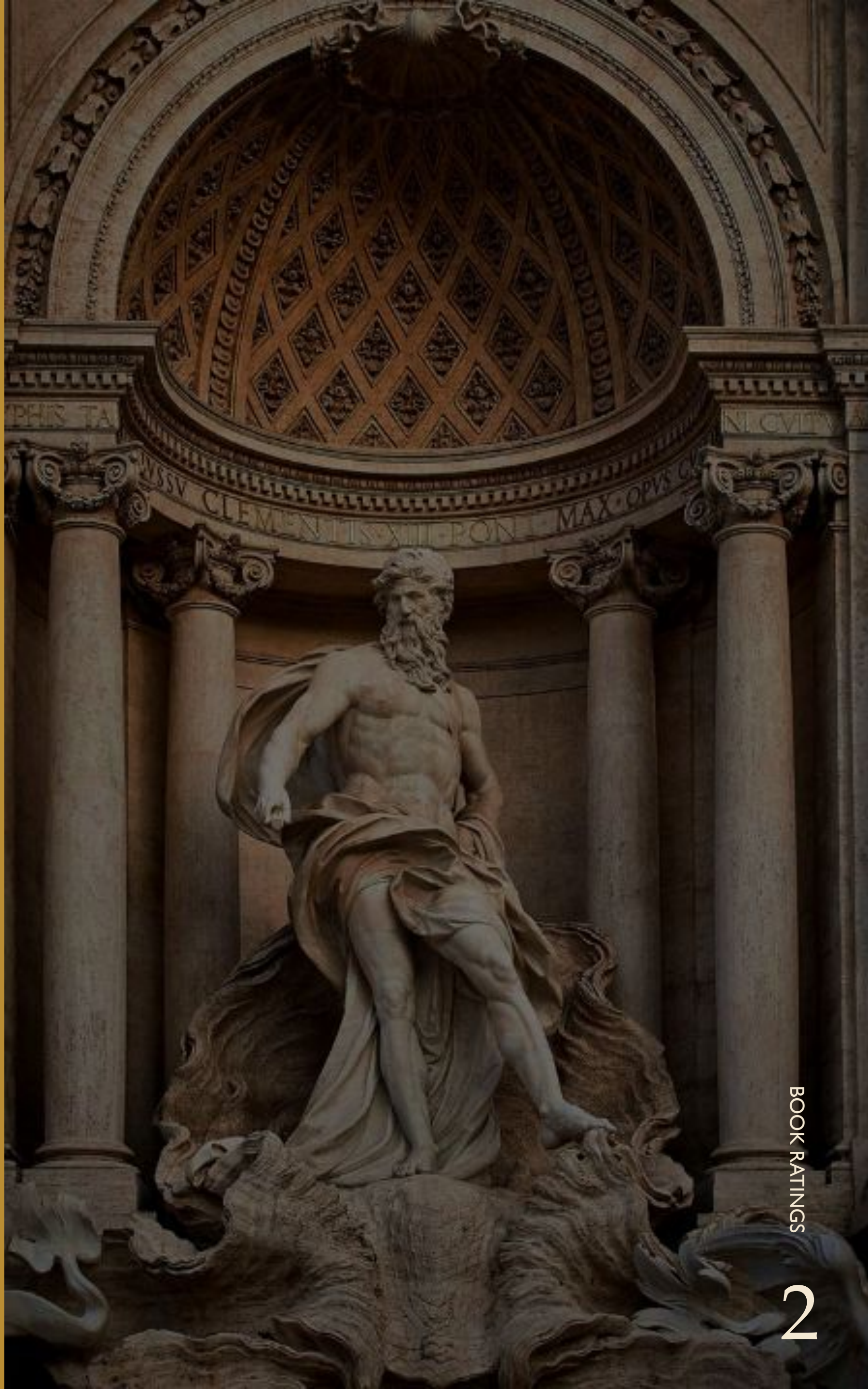
Unveiling Literary Insights

HARNESSING THE POWER OF DATA FOR PREDICTIVE MODELING OF BOOK RATINGS THROUGH LINEAR REGRESSION

By Abdulla Mashaly, Jeremy Magee, Akhil Karandikar, and, Lori Girton

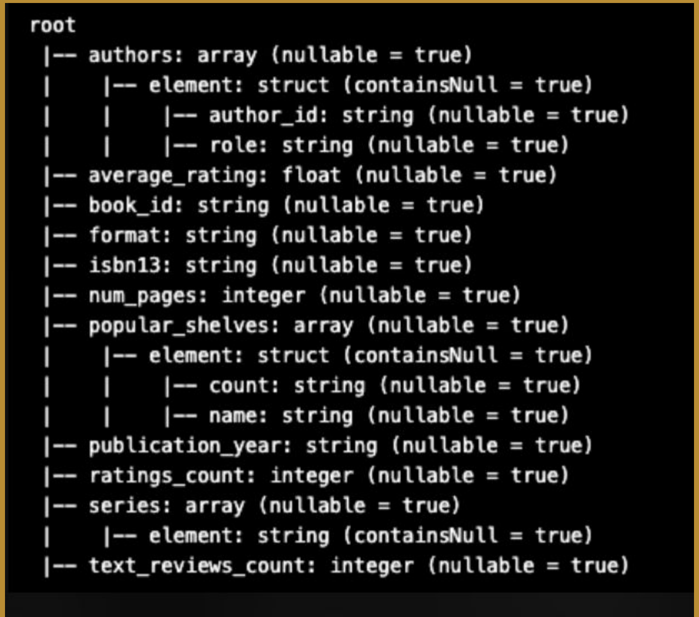
Our Epic Odyssey

- Our first thought was to create a prediction engine, one that takes input from the user and outputs recommended books. Soon we realized some of the issues with such an approach
  - One issue was that we could not measure accuracy of the predictions
- We pivoted to predicting book ratings given the multitude of data our APIs were giving us.
- From there we tried several models and tweaks before using a linear regression learning model.
- In tandem with that we tried a random forest classification, and it worked! There was much rejoicing.
- Then we finally began our optimization process.



Let’s look at some Sample Data!

- Our API was from Goodreads (via Kaggle), an open data source with multitudes of information about a given book.
- This dataset includes everything about each book from ISBN, average author ratings, number of times a book appears on a to-read list, number of pages, etc.



```
root
 |-- authors: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- author_id: string (nullable = true)
 |    |    |-- role: string (nullable = true)
 |-- average_rating: float (nullable = true)
 |-- book_id: string (nullable = true)
 |-- format: string (nullable = true)
 |-- isbn13: string (nullable = true)
 |-- num_pages: integer (nullable = true)
 |-- popular_shelves: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- count: string (nullable = true)
 |    |    |-- name: string (nullable = true)
 |-- publication_year: string (nullable = true)
 |-- ratings_count: integer (nullable = true)
 |-- series: array (nullable = true)
 |    |-- element: string (containsNull = true)
 |-- text_reviews_count: integer (nullable = true)
```

Data Collection and Cleaning

PYSPARK AND SQL!

- Our initial dataset had a size of 2 GB, which was unwieldy for the computing power and memory we have on hand.
- PySpark was irreplaceable for caching our info into a manageable dataframe so that we could decide on necessary features for our model.

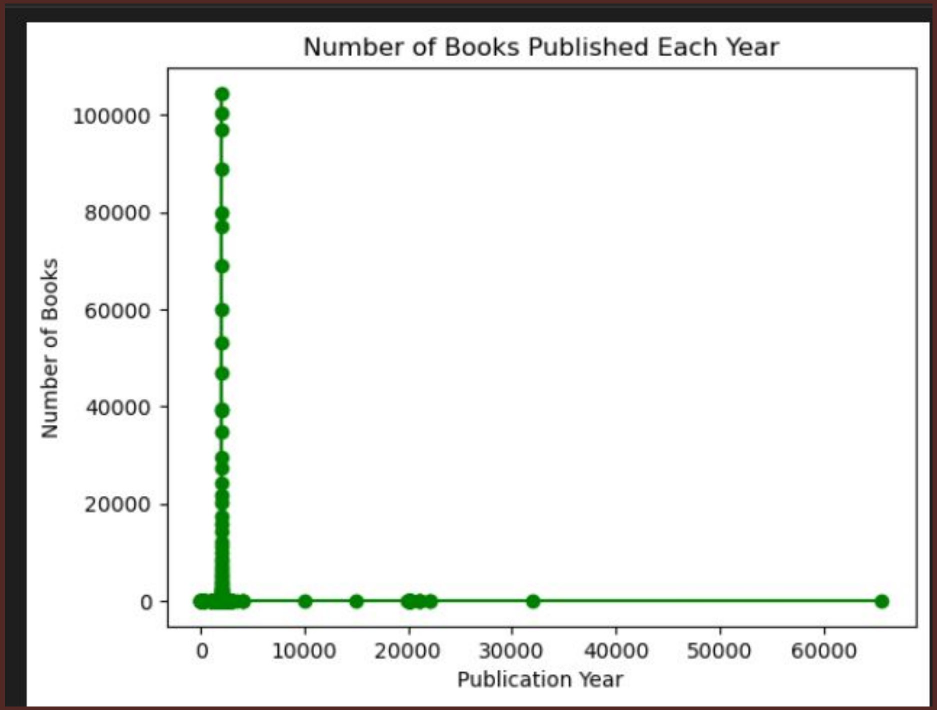
DATA CLEANING:
WHAT THE HECK IS THAT?!

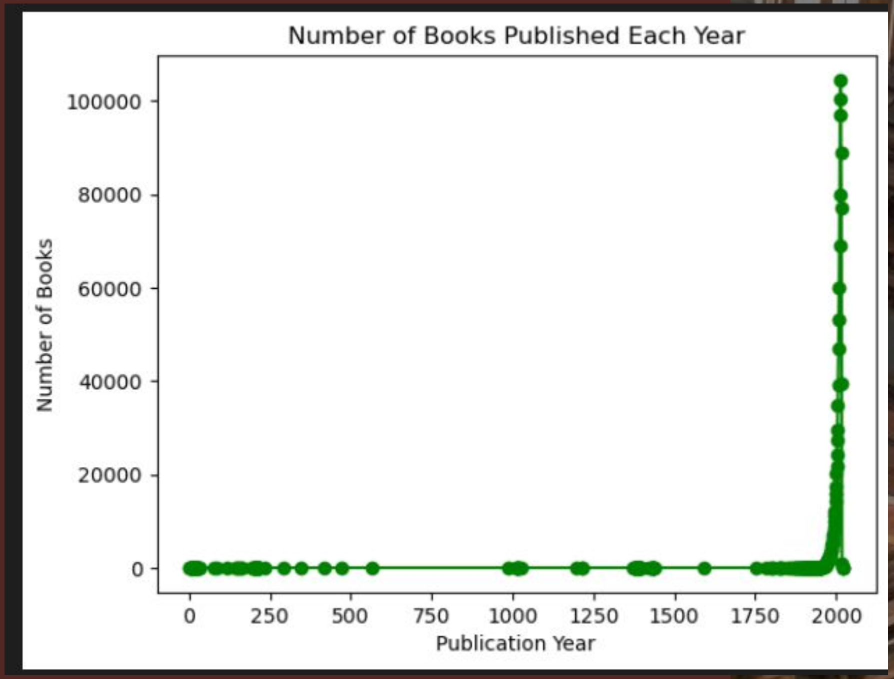
- Data cleaning is necessary for the proper EDA needed to make a model
- We cleaned the data and found many curious and odd entries:
- Some examples:

Page Count	Published Year	Ratings
• 0 • 8,345	• 5 (yes, 5) • 56204	• 0 (that’s just mean)

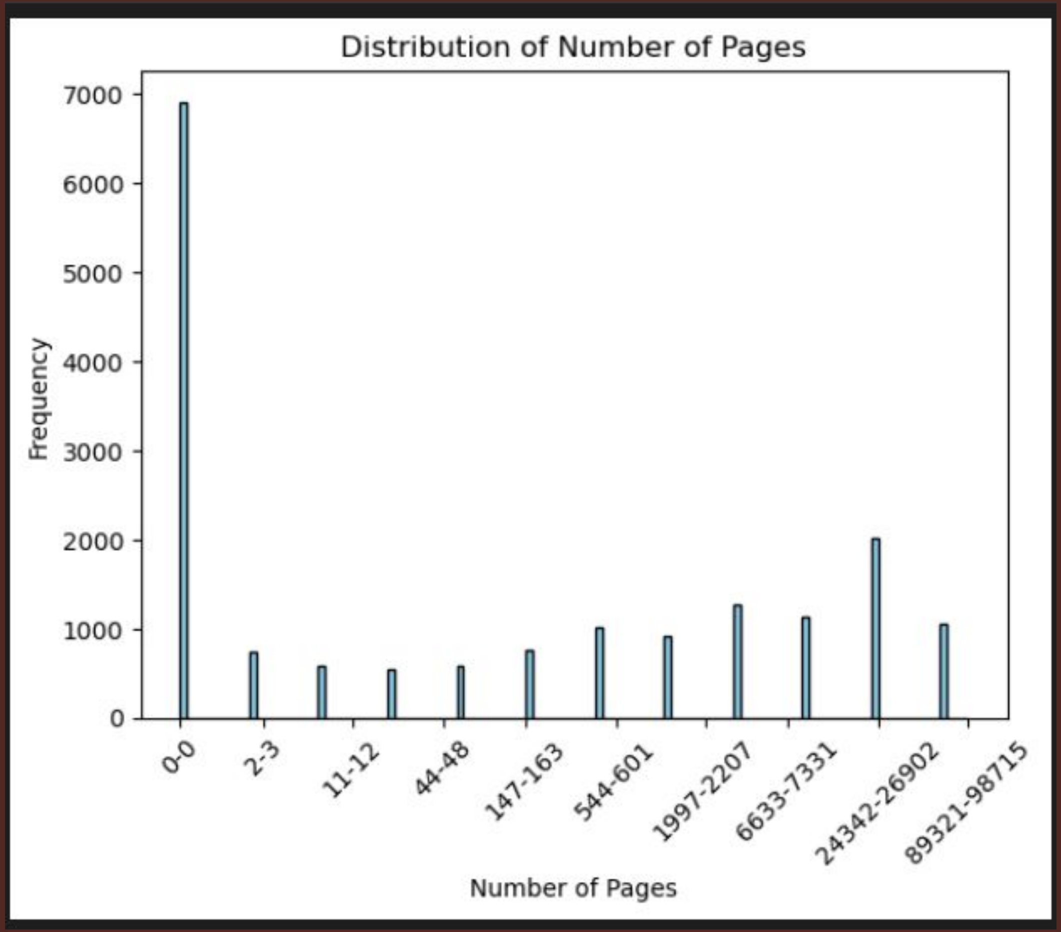
Data Exploration & Analysis

EDA
Step 1

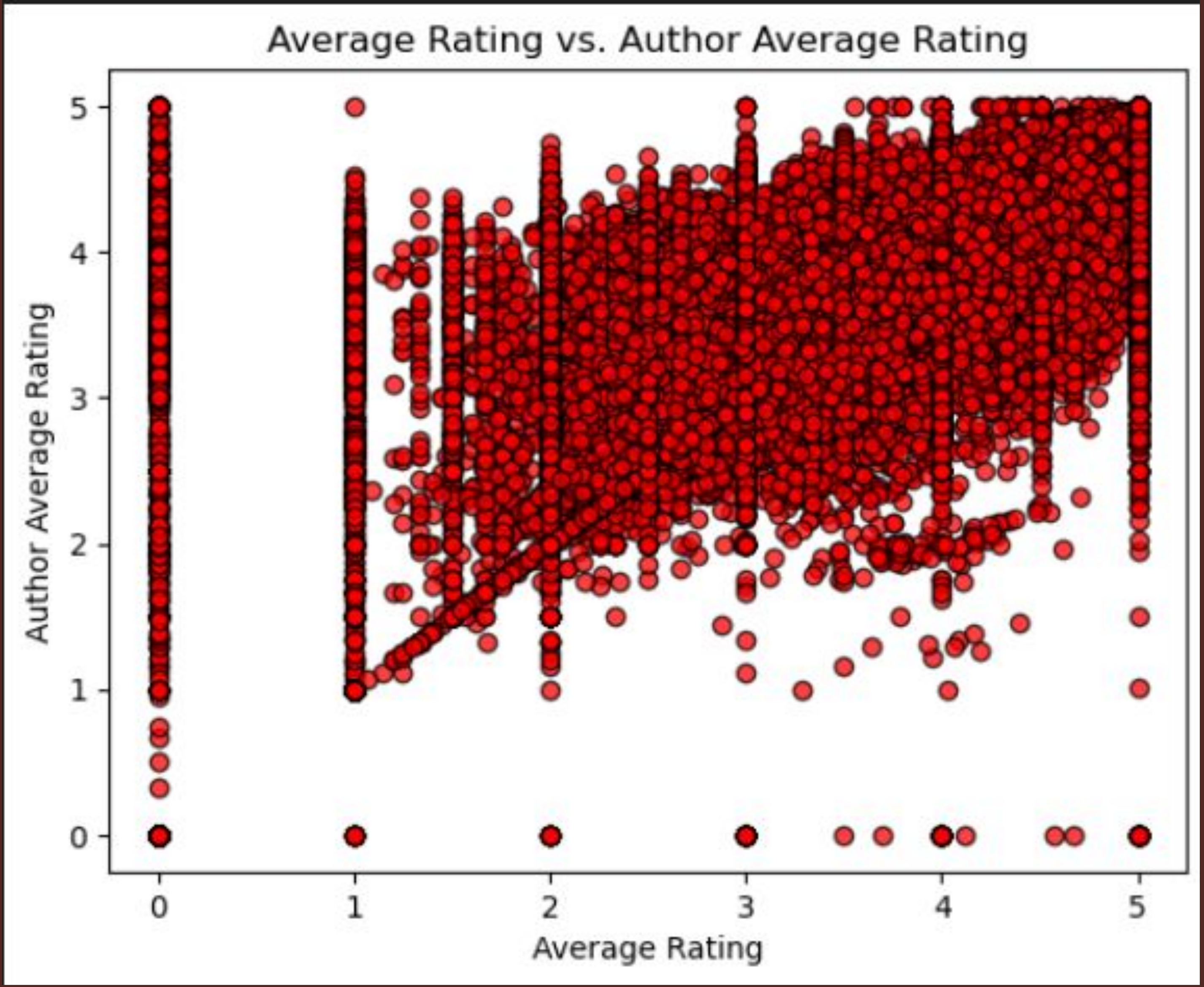

Number of Books Published Each Year
Number of Books
100000
80000
60000
40000
20000
0
0
10000
20000
30000
40000
50000
60000
Publication Year


Number of Books Published Each Year
Number of Books
100000
80000
60000
40000
20000
0
0
250
500
750
1000
1250
1500
1750
2000
Publication Year

EDA Step 1

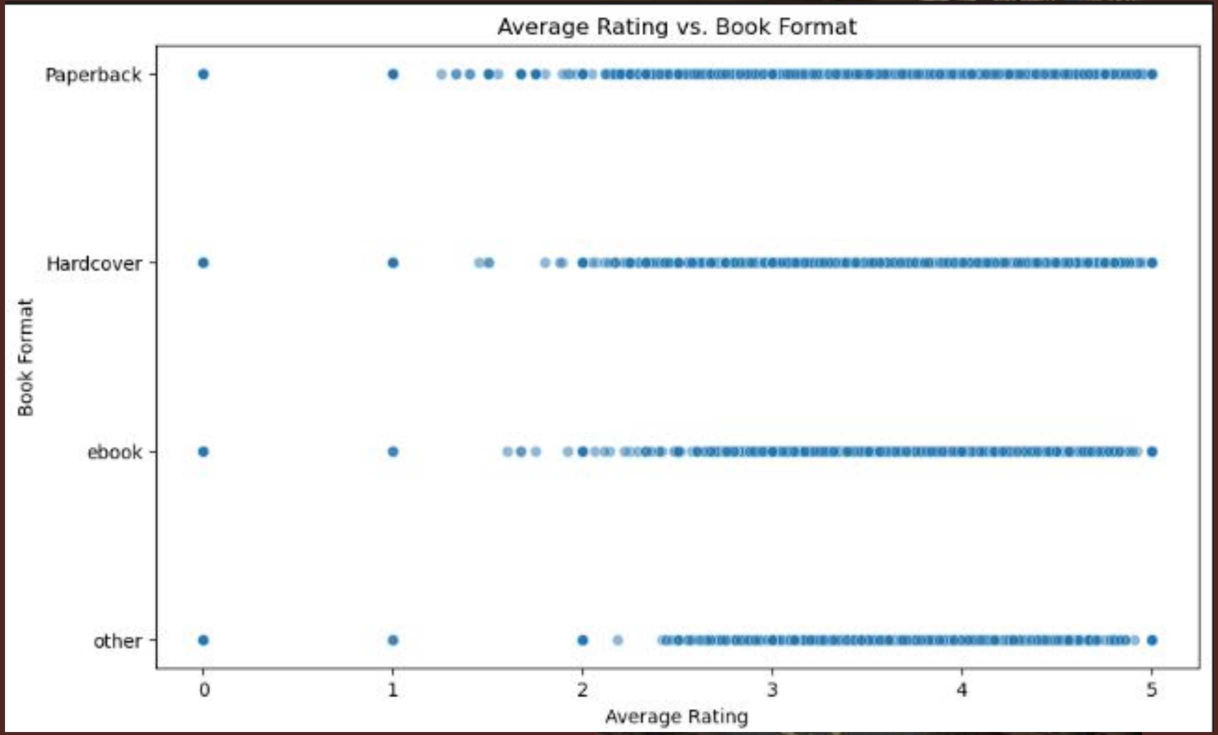

Distribution of Number of Pages

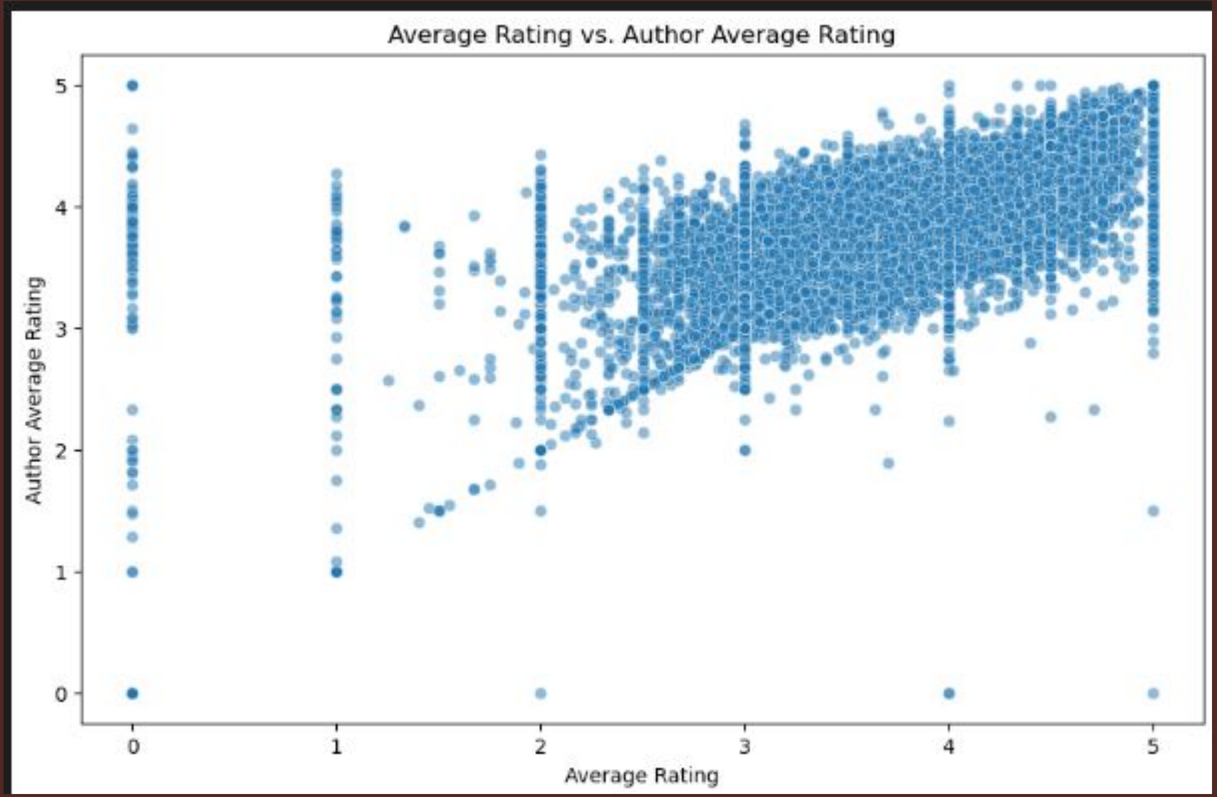
EDA Step 2


Average Rating vs. Author Average Rating
Author Average Rating
Average Rating
0
1
2
3
4
5
0
1
2
3
4
5

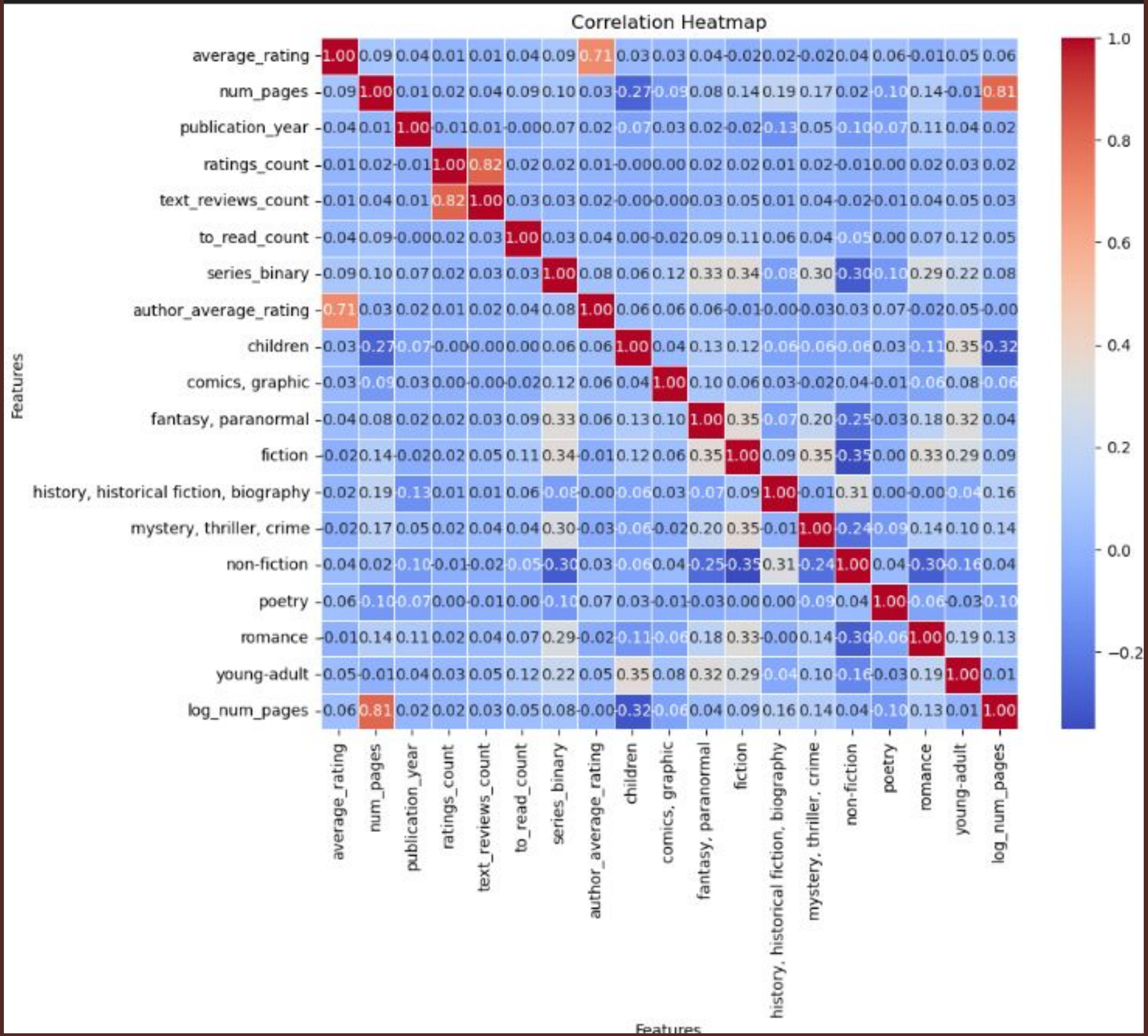
EX DONO

EDA: Sample 3 Data





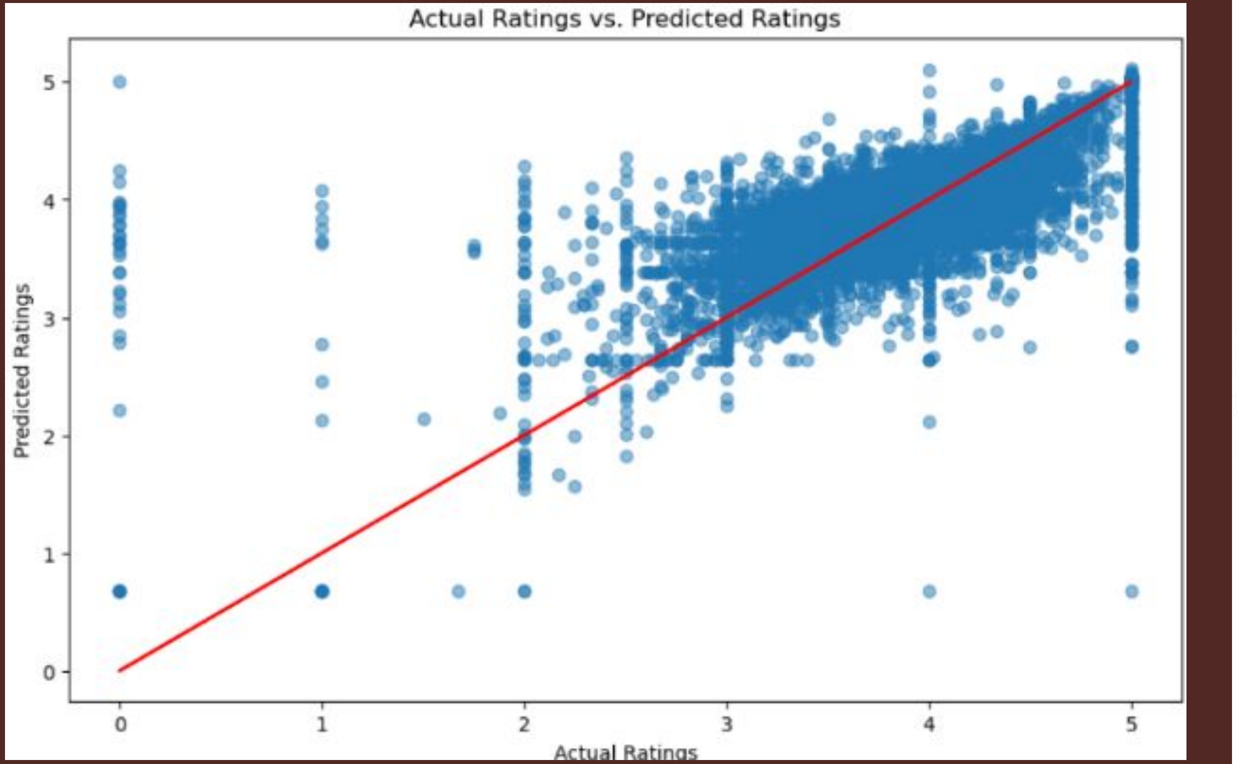
EDA: Sample 3 Data

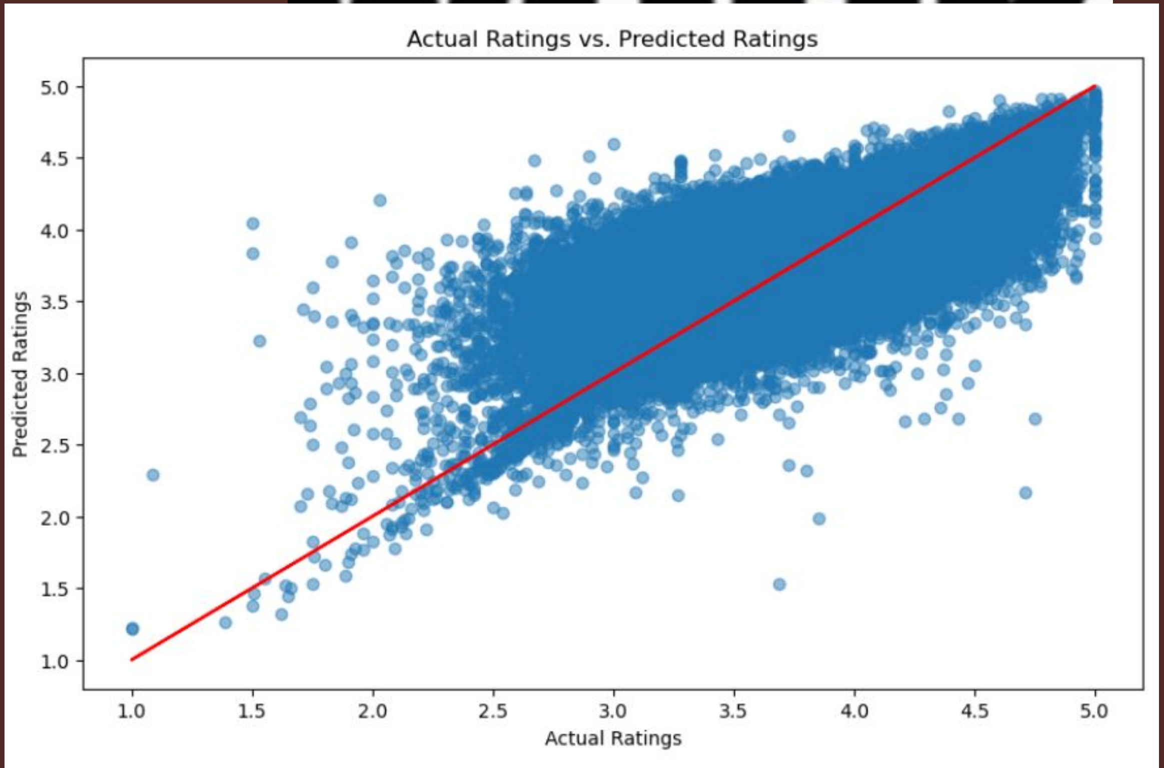


	average_rating	num_pages	publication_year	ratings_count	text_reviews_count	to_read_count	series_binary	author_average_rating	children	comics, graphic	fantasy, paranormal	fiction	history, historical fiction, biography	mystery, thriller, crime	non-fiction	poetry	romance	young-adult	log_num_pages
average_rating	1.00	0.09	0.04	0.01	0.01	0.04	0.09	0.71	0.03	0.03	0.04	-0.02	0.02	-0.02	0.04	0.06	-0.01	0.05	0.06
num_pages	0.09	1.00	0.01	0.02	0.04	0.09	0.10	0.03	-0.27	-0.09	0.08	0.14	0.19	0.17	0.02	-0.10	0.14	-0.01	0.81
publication_year	0.04	0.01	1.00	-0.01	0.01	-0.00	0.07	0.02	-0.07	0.03	0.02	-0.02	-0.13	0.05	-0.10	-0.07	0.11	0.04	0.02
ratings_count	0.01	0.02	-0.01	1.00	0.82	0.02	0.02	0.01	-0.00	0.00	0.02	0.02	0.01	0.02	-0.01	0.00	0.02	0.03	0.02
text_reviews_count	0.01	0.04	0.01	0.82	1.00	0.03	0.03	0.02	-0.00	-0.00	0.03	0.05	0.01	0.04	-0.02	-0.01	0.04	0.05	0.03
to_read_count	0.04	0.09	-0.00	0.02	0.03	1.00	0.03	0.04	0.00	-0.02	0.09	0.11	0.06	0.04	-0.05	0.00	0.07	0.12	0.05
series_binary	0.09	0.10	0.07	0.02	0.03	0.03	1.00	0.08	0.06	0.12	0.33	0.34	-0.08	0.30	-0.30	-0.10	0.29	0.22	0.08
author_average_rating	0.71	0.03	0.02	0.01	0.02	0.04	0.08	1.00	0.06	0.06	0.06	-0.01	-0.00	-0.03	0.03	0.07	-0.02	0.05	-0.00
children	0.03	-0.27	-0.07	-0.00	0.00	0.00	0.06	0.06	1.00	0.04	0.13	0.12	-0.06	-0.06	-0.06	0.03	-0.11	0.35	-0.32
comics, graphic	0.03	-0.09	0.03	0.00	-0.00	-0.02	0.12	0.06	0.04	1.00	0.10	0.06	0.03	-0.02	0.04	-0.01	-0.06	0.08	-0.06
fantasy, paranormal	0.04	0.08	0.02	0.02	0.03	0.09	0.33	0.06	0.13	0.10	1.00	0.35	-0.07	0.20	-0.25	-0.03	0.18	0.32	0.04
fiction	-0.02	0.14	-0.02	0.02	0.05	0.11	0.34	-0.01	0.12	0.06	0.35	1.00	0.09	0.35	-0.35	0.00	0.33	0.29	0.09
history, historical fiction, biography	0.02	0.19	-0.13	0.01	0.01	0.06	-0.08	-0.00	-0.06	0.03	-0.07	0.09	1.00	-0.01	0.31	0.00	-0.00	-0.04	0.16
mystery, thriller, crime	-0.02	0.17	0.05	0.02	0.04	0.04	0.30	-0.03	-0.06	-0.02	0.20	0.35	-0.01	1.00	-0.24	-0.09	0.14	0.10	0.14
non-fiction	0.04	0.02	-0.10	-0.01	-0.02	-0.05	-0.30	0.03	-0.06	0.04	-0.25	-0.35	0.31	-0.24	1.00	0.04	-0.30	-0.16	0.04
poetry	0.06	-0.10	-0.07	0.00	-0.01	0.00	-0.10	0.07	0.03	-0.01	-0.03	0.00	0.00	-0.09	0.04	1.00	-0.06	-0.03	-0.10
romance	-0.01	0.14	0.11	0.02	0.04	0.07	0.29	-0.02	-0.11	-0.06	0.18	0.33	-0.00	0.14	-0.30	-0.06	1.00	0.19	0.13
young-adult	0.05	-0.01	0.04	0.03	0.05	0.12	0.22	0.05	0.35	0.08	0.32	0.29	-0.04	0.10	-0.16	-0.03	0.19	1.00	0.01
log_num_pages	0.06	0.81	0.02	0.02	0.03	0.05	0.08	-0.00	-0.32	-0.06	0.04	0.09	0.16	0.14	0.04	-0.10	0.13	0.01	1.00

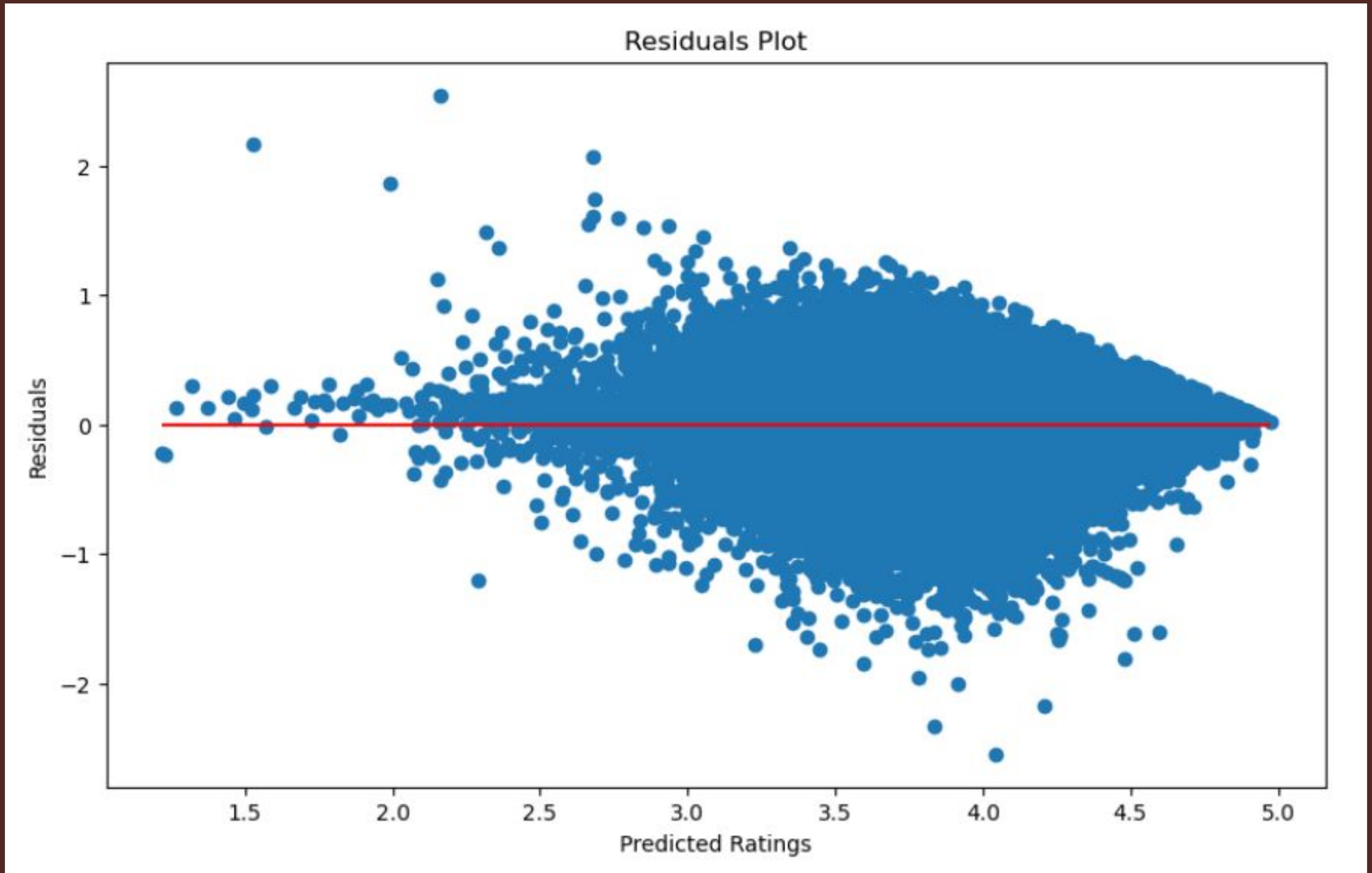
Model Optimization & Selection

Model 1: Neural Network

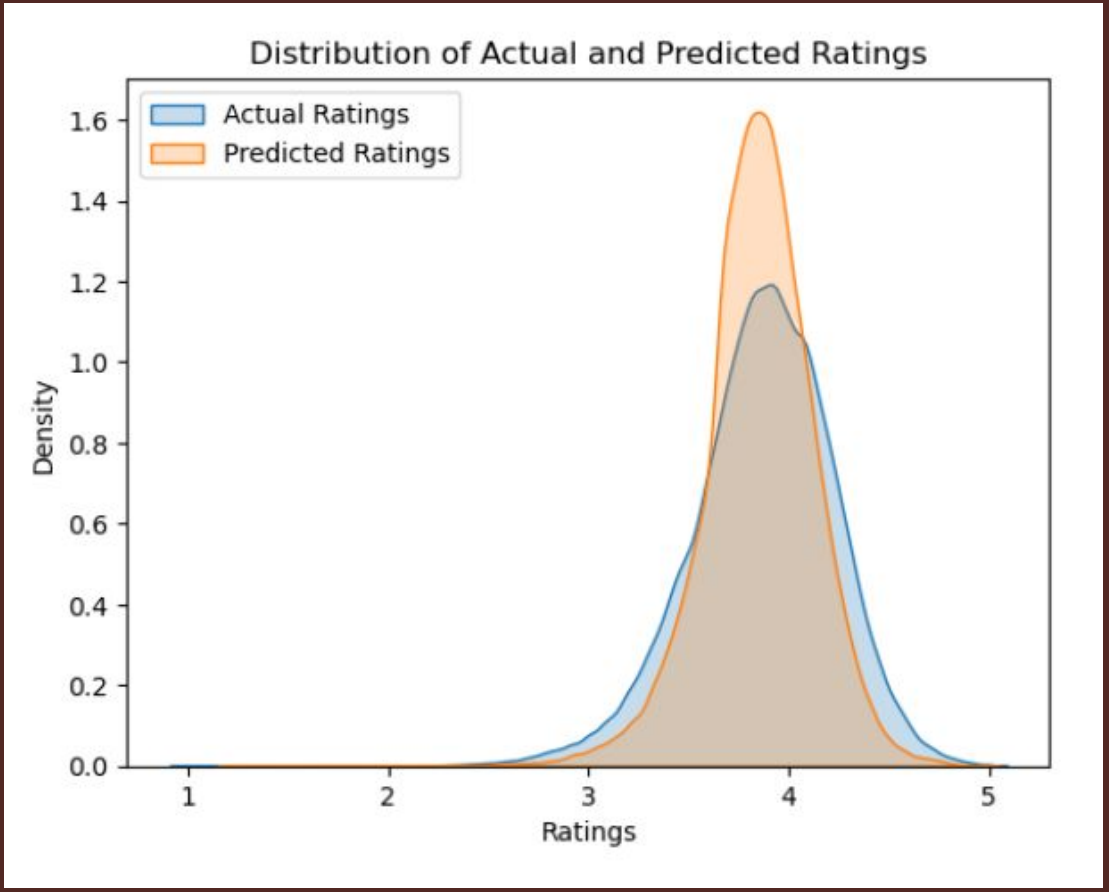




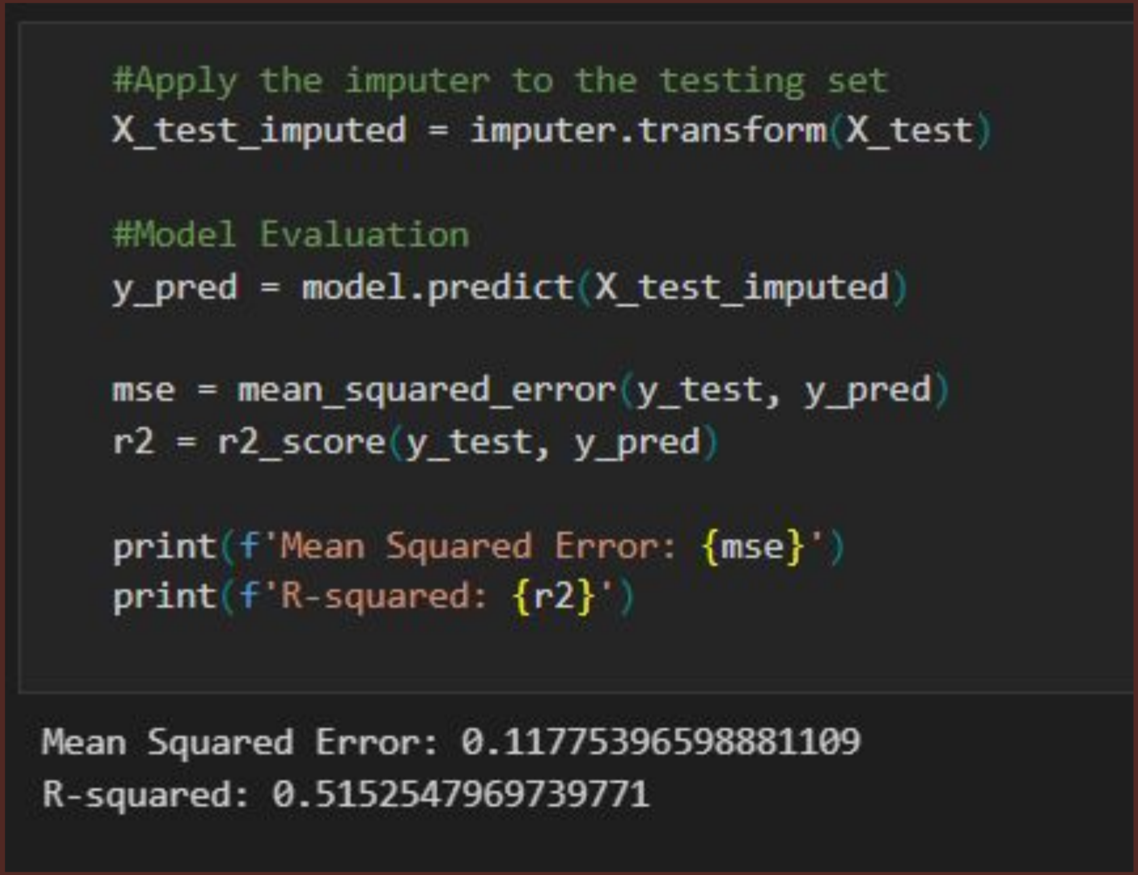
Model 1: Neural Network



Model 1: Neural Network



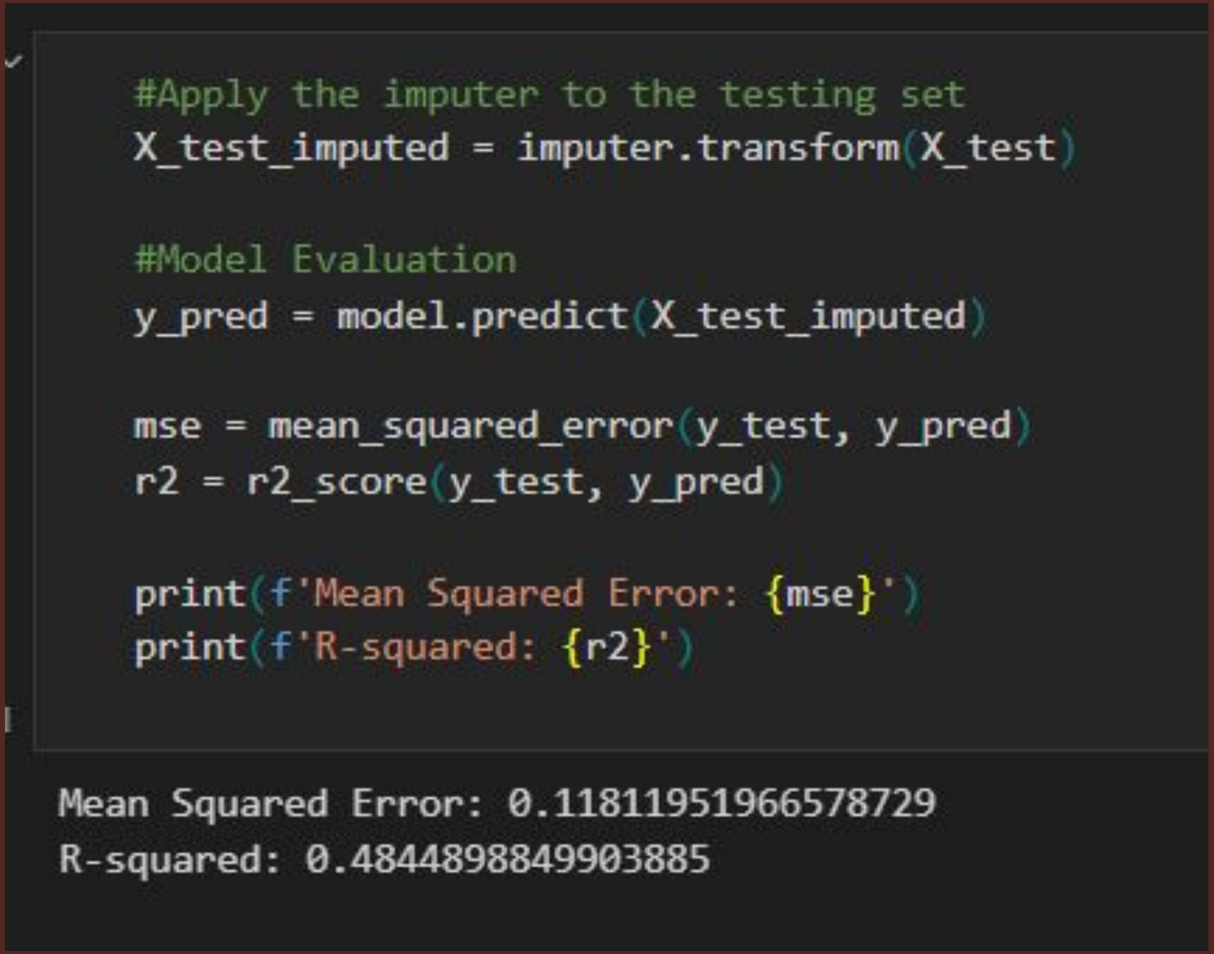
Model 2:
Linear Regression



```
#Apply the imputer to the testing set
X_test_imputed = imputer.transform(X_test)

#Model Evaluation
y_pred = model.predict(X_test_imputed)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

```
Mean Squared Error: 0.11775396598881109
R-squared: 0.5152547969739771
```

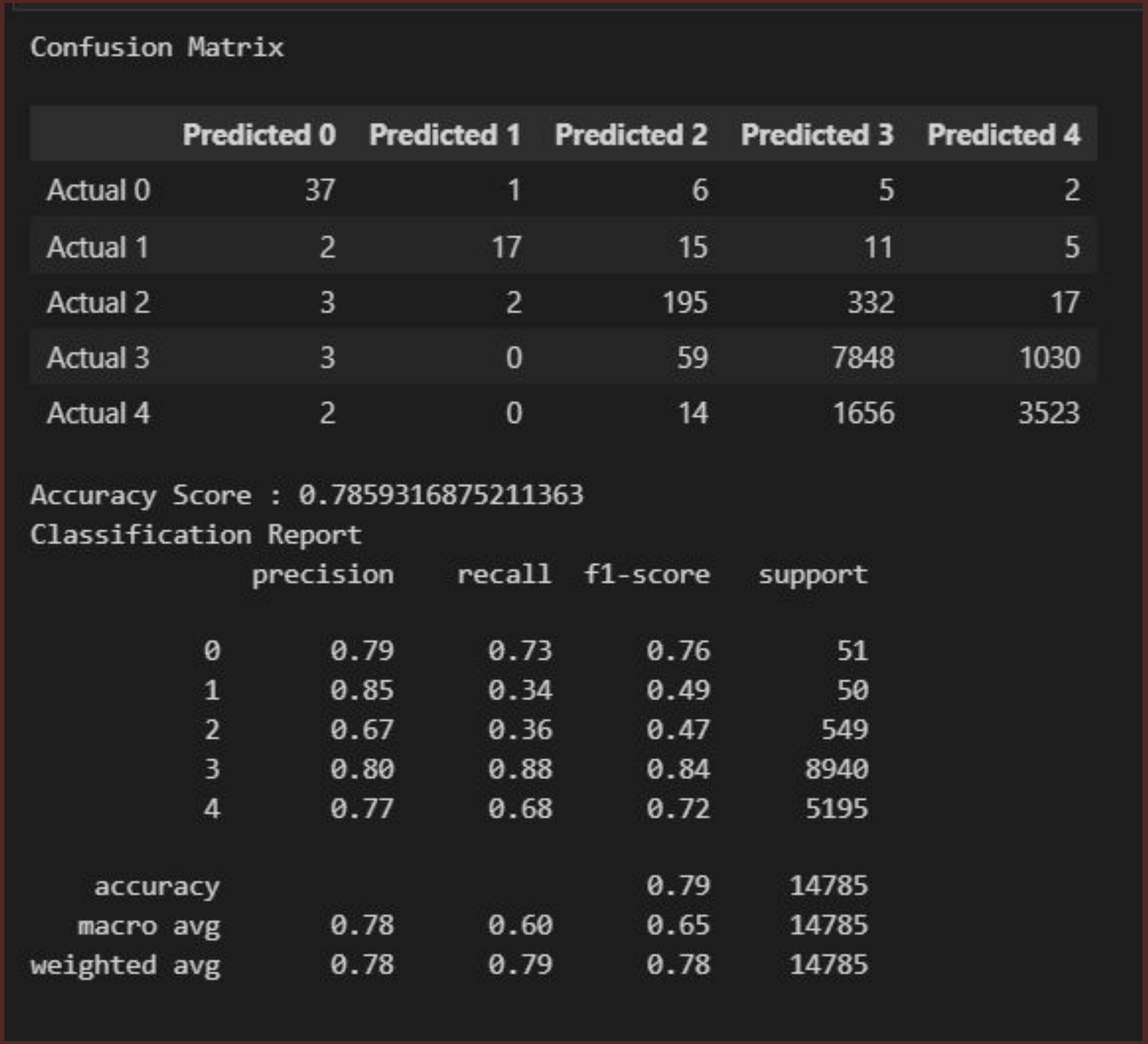
Model 2:
Linear Regression (Again)



```
#Apply the imputer to the testing set
X_test_imputed = imputer.transform(X_test)

#Model Evaluation
y_pred = model.predict(X_test_imputed)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

```
Mean Squared Error: 0.11811951966578729
R-squared: 0.4844898849903885
```

Model 3: Random Forest



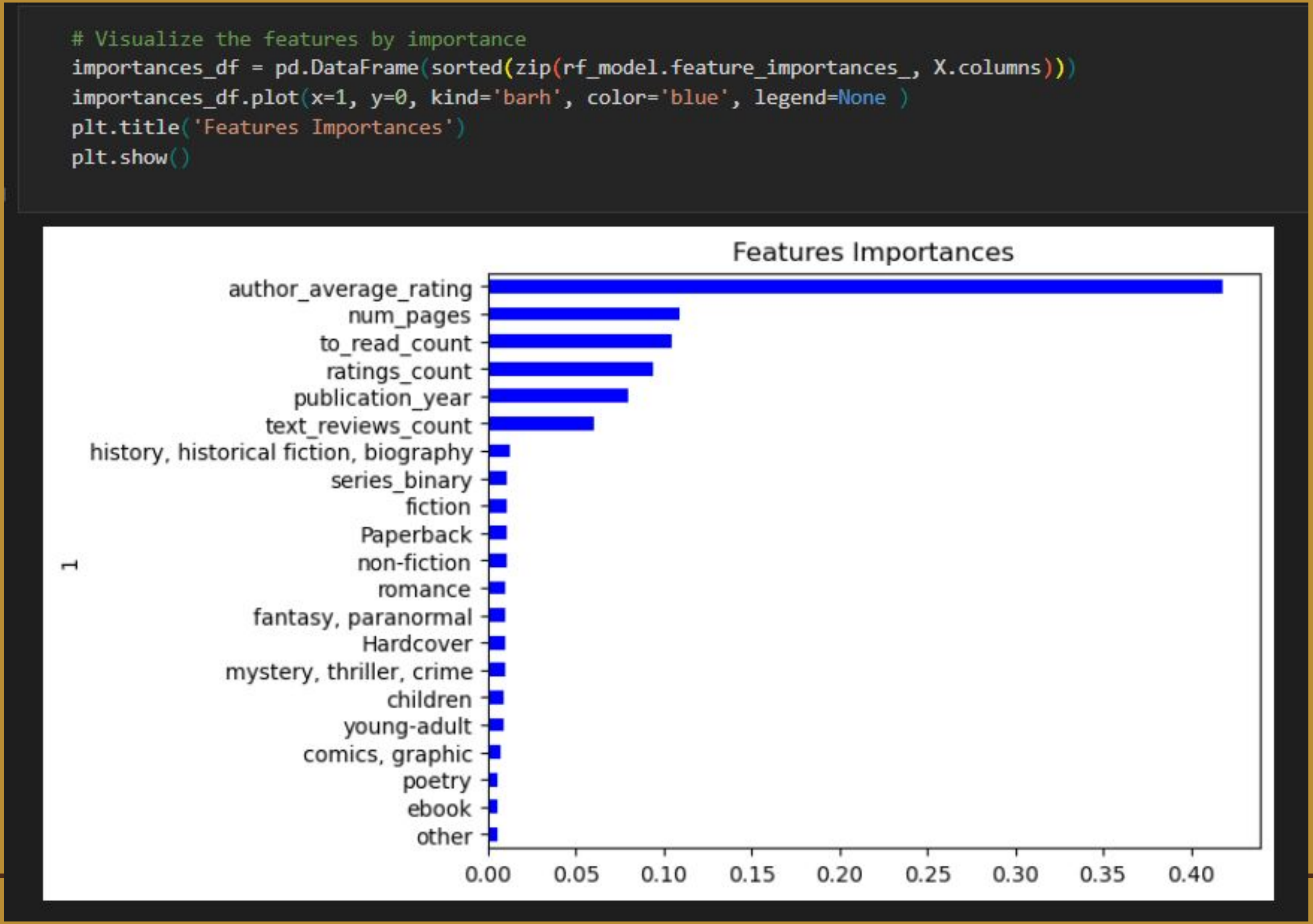
Confusion Matrix

	Predicted 0	Predicted 1	Predicted 2	Predicted 3	Predicted 4
Actual 0	37	1	6	5	2
Actual 1	2	17	15	11	5
Actual 2	3	2	195	332	17
Actual 3	3	0	59	7848	1030
Actual 4	2	0	14	1656	3523

```
Accuracy Score : 0.7859316875211363
Classification Report
              precision    recall  f1-score   support

           0       0.79      0.73      0.76        51
           1       0.85      0.34      0.49        50
           2       0.67      0.36      0.47       549
           3       0.80      0.88      0.84      8940
           4       0.77      0.68      0.72      5195

    accuracy                           0.79     14785
   macro avg       0.78      0.60      0.65     14785
weighted avg       0.78      0.79      0.78     14785
```

Random Forest Model:

Feature Rankings



```
# Visualize the features by importance
importances_df = pd.DataFrame(sorted(zip(rf_model.feature_importances_, X.columns)))
importances_df.plot(x=1, y=0, kind='barh', color='blue', legend=None )
plt.title('Features Importances')
plt.show()
```

> “BUT YOU, BRAVE AND ADEPT FROM THIS DAY ON . . . THERE’S HOPE THAT YOU WILL REACH YOUR GOAL . . . THE JOURNEY THAT STIRS YOU NOW IS NOT FAR OFF.”

- HOMER, THE ODYSSEY

CITATIONS

ARTICLES

- Mengting Wan, Julian McAuley, “Item Recommendation on Monotonic Behavior Chains”, in RecSys’18 [bibtex}
- Mengting Wan, Rishabh Misra, Ndapa Nakashole, Julian McAuley, “Fine-Grained Spoiler Detection from Large-Scale Review Corpora”, in ACL’19 [bibtex]

DATA

- Wan, Mengting. (2023).goodreads.GitHub.https://github.com/MengtingWan/goodreads
- Ahmad. (2023, October). Goodreads Book Reviews, Version 1. Retrieved November 22, 2023 from https://www.kaggle.com/datasets/pypiahmad/goodreads-book-reviews1

ALRIGHT DUDE, LET'S GET THE HELL OUTTA HERE