Predict book rating

JosueARz

The purpose of this project is to predict the rating of a book and see what are the important characteristics and factors why some books are more popular than others.

We are going to treat this problem with some differences in the preprocessing of the data, specifically for outliers, of which we will see what the differences are for each choice of treatment of our data.

Import libraries

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

from sklearn import preprocessing
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn import metrics
```

Read data

[2]:
```
data = pd.read_csv('books.csv', error_bad_lines=False)
```

```
C:\ProgramData\Anaconda3\lib\site-
packages\IPython\core\interactiveshell.py:3441: FutureWarning: The
error_bad_lines argument has been deprecated and will be removed in a␣
 ↪future
version.


  exec(code_obj, self.user_global_ns, self.user_ns)
b'Skipping line 3350: expected 12 fields, saw 13\nSkipping line 4704:␣
 ↪expected
12 fields, saw 13\nSkipping line 5879: expected 12 fields, saw␣
 ↪13\nSkipping line
```

```
8981: expected 12 fields, saw 13\n'
```

Data Exploration

[3]:
```
data.head()
```

[3]:
```
   bookID                                              title  \
0       1  Harry Potter and the Half-Blood Prince (Harry ...
1       2  Harry Potter and the Order of the Phoenix (Har...
2       4  Harry Potter and the Chamber of Secrets (Harry...
3       5  Harry Potter and the Prisoner of Azkaban (Harr...
4       8  Harry Potter Boxed Set  Books 1-5 (Harry Potte...

                    authors  average_rating        isbn        ␣
 ↪isbn13  \
0  J.K. Rowling/Mary GrandPré            4.57  0439785960  9780439785969
1  J.K. Rowling/Mary GrandPré            4.49  0439358078  9780439358071
2               J.K. Rowling            4.42  0439554896  9780439554893
3  J.K. Rowling/Mary GrandPré            4.56  043965548X  9780439655484
4  J.K. Rowling/Mary GrandPré            4.78  0439682584  9780439682589

  language_code  num_pages  ratings_count  text_reviews_count  \
0           eng        652        2095690               27591
1           eng        870        2153167               29221
2           eng        352           6333                 244
3           eng        435        2339585               36325
4           eng       2690          41428                 164

  publication_date        publisher
0        9/16/2006  Scholastic Inc.
1         9/1/2004  Scholastic Inc.
2        11/1/2003       Scholastic
3         5/1/2004  Scholastic Inc.
4        9/13/2004       Scholastic
```

[4]:
```
data.shape
```

[4]:
```
(11123, 12)
```

[5]:
```
data.describe()
```

[5]:
```
              bookID  average_rating        isbn13     num_pages  \
count   11123.000000    11123.000000  1.112300e+04  11123.000000
mean    21310.856963        3.934075  9.759880e+12    336.405556
std     13094.727252        0.350485  4.429758e+11    241.152626
min         1.000000        0.000000  8.987060e+09      0.000000
25%     10277.500000        3.770000  9.780345e+12    192.000000
```

```
50%    20287.000000        3.960000  9.780582e+12    299.000000
75%    32104.500000        4.140000  9.780872e+12    416.000000
max    45641.000000        5.000000  9.790008e+12   6576.000000

       ratings_count  text_reviews_count
count   1.112300e+04        11123.000000
mean    1.794285e+04          542.048099
std     1.124992e+05         2576.619589
min     0.000000e+00            0.000000
25%     1.040000e+02            9.000000
50%     7.450000e+02           47.000000
75%     5.000500e+03          238.000000
max     4.597666e+06        94265.000000
```

[6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11123 entries, 0 to 11122
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   bookID              11123 non-null  int64
 1   title               11123 non-null  object
 2   authors             11123 non-null  object
 3   average_rating      11123 non-null  float64
 4   isbn                11123 non-null  object
 5   isbn13              11123 non-null  int64
 6   language_code       11123 non-null  object
 7     num_pages         11123 non-null  int64
 8   ratings_count       11123 non-null  int64
 9   text_reviews_count  11123 non-null  int64
 10  publication_date    11123 non-null  object
 11  publisher           11123 non-null  object
dtypes: float64(1), int64(5), object(6)
memory usage: 1.0+ MB
```

[7]:
```
#searching for null values
data.isnull().any()
```

[7]:
```
bookID              False
title               False
authors             False
average_rating      False
isbn                False
isbn13              False
language_code       False
  num_pages         False
ratings_count       False
```

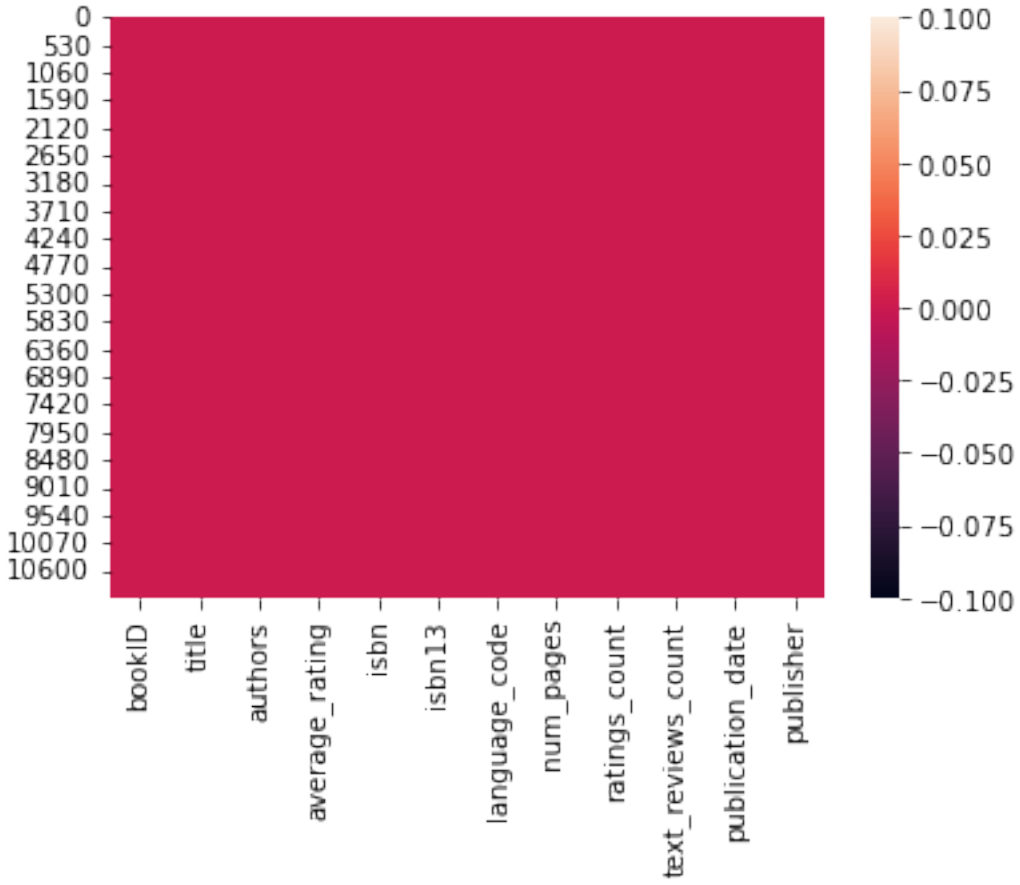
```
text_reviews_count    False
publication_date      False
publisher             False
dtype: bool
```

[8]:
```
#searching for duplicate values
data.duplicated().any()
```

[8]:
```
False
```

[9]:
```
#let´s to visualize the previous result
sns.heatmap(data.isnull())
```

[9]:
```
<AxesSubplot:>
```



[10]:
```
#numbers of books per rating
sns.barplot(data['average_rating'].value_counts().head(20).index,␣
 ↪data['average_rating'].value_counts().head(20))
plt.title('Number of Books Each Rating Received\n')
plt.xlabel('Ratings')
plt.ylabel('Counts')
```

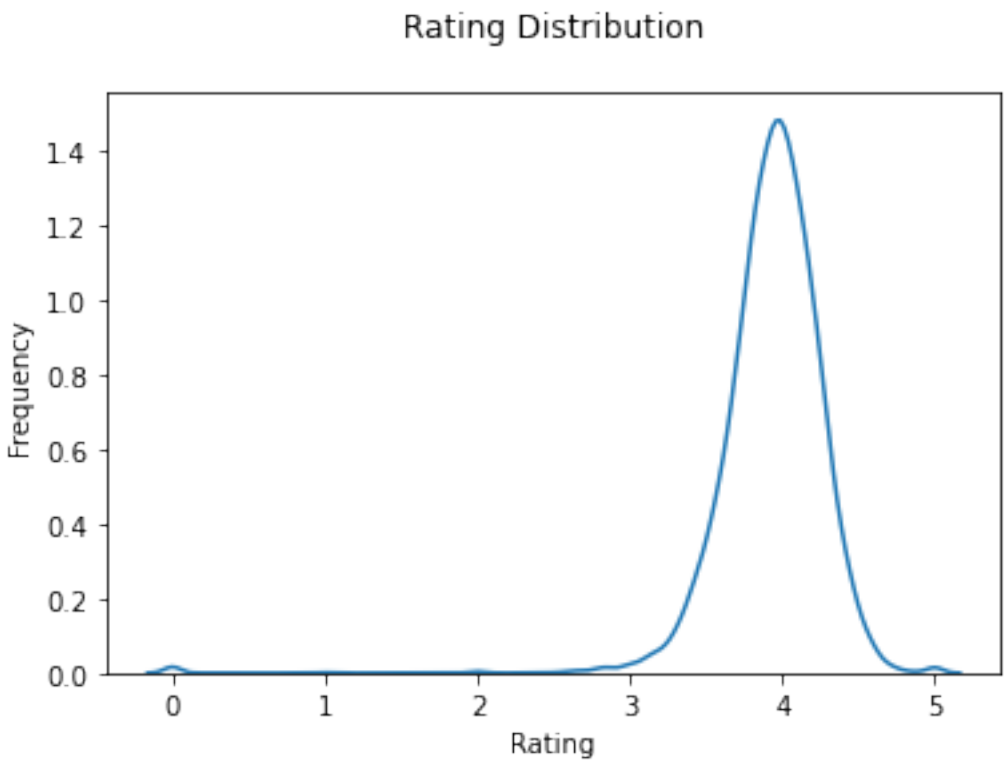
```
plt.xticks(rotation=90)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From␣
  ↪version
0.12, the only valid positional argument will be `data`, and passing␣
  ↪other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[10]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15,␣
  ↪16,
        17, 18, 19]),
 [Text(0, 0, '3.82'),
  Text(1, 0, '3.83'),
  Text(2, 0, '3.88'),
  Text(3, 0, '3.89'),
  Text(4, 0, '3.9'),
  Text(5, 0, '3.92'),
  Text(6, 0, '3.93'),
  Text(7, 0, '3.94'),
  Text(8, 0, '3.96'),
  Text(9, 0, '3.97'),
  Text(10, 0, '3.98'),
  Text(11, 0, '3.99'),
  Text(12, 0, '4.0'),
  Text(13, 0, '4.02'),
  Text(14, 0, '4.04'),
  Text(15, 0, '4.05'),
  Text(16, 0, '4.06'),
  Text(17, 0, '4.07'),
  Text(18, 0, '4.09'),
  Text(19, 0, '4.17')])
```

[11]:
```
# ratings distribution
sns.kdeplot(data['average_rating'])
plt.title('Rating Distribution\n')
plt.xlabel('Rating')
plt.ylabel('Frequency')
```

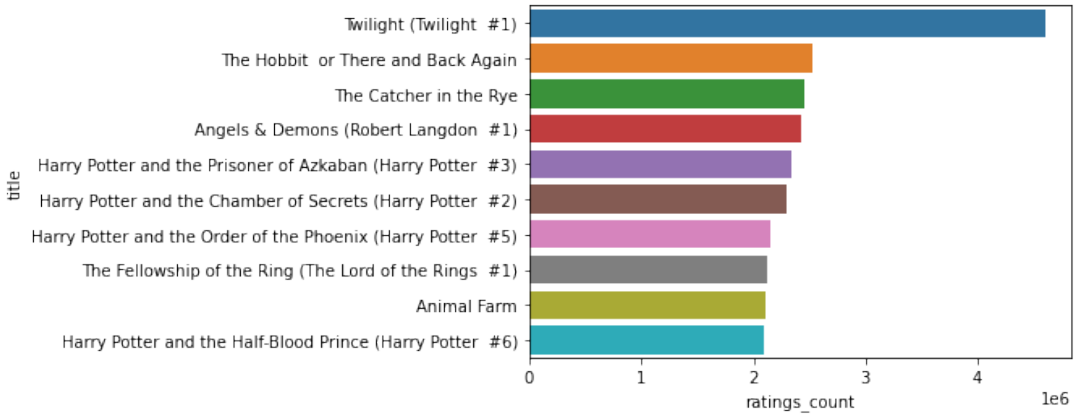
[11]:
```
Text(0, 0.5, 'Frequency')
```



[12]:
```
# highest rated books
popular_books = data.nlargest(10, ['ratings_count']).
 ↪set_index('title')['ratings_count']
sns.barplot(popular_books, popular_books.index)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From␣
 ↪version
0.12, the only valid positional argument will be `data`, and passing␣
 ↪other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

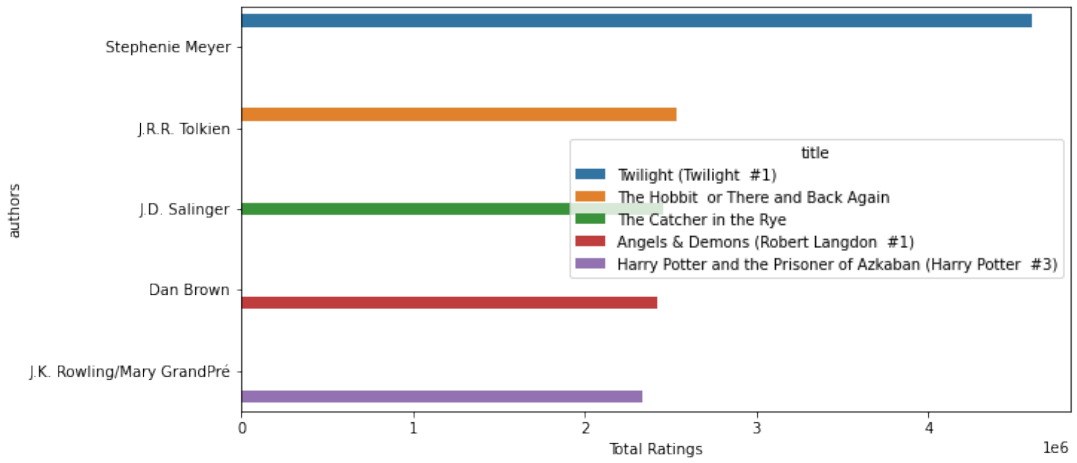
[12]: <AxesSubplot:xlabel='ratings_count', ylabel='title'>



[13]:
```
# authors with highest rated books
plt.figure(figsize=(10, 5))
authors = data.nlargest(5, ['ratings_count']).set_index('authors')
sns.barplot(authors['ratings_count'], authors.index, ci = None, hue =␣
 ↪authors['title'])
plt.xlabel('Total Ratings')
```

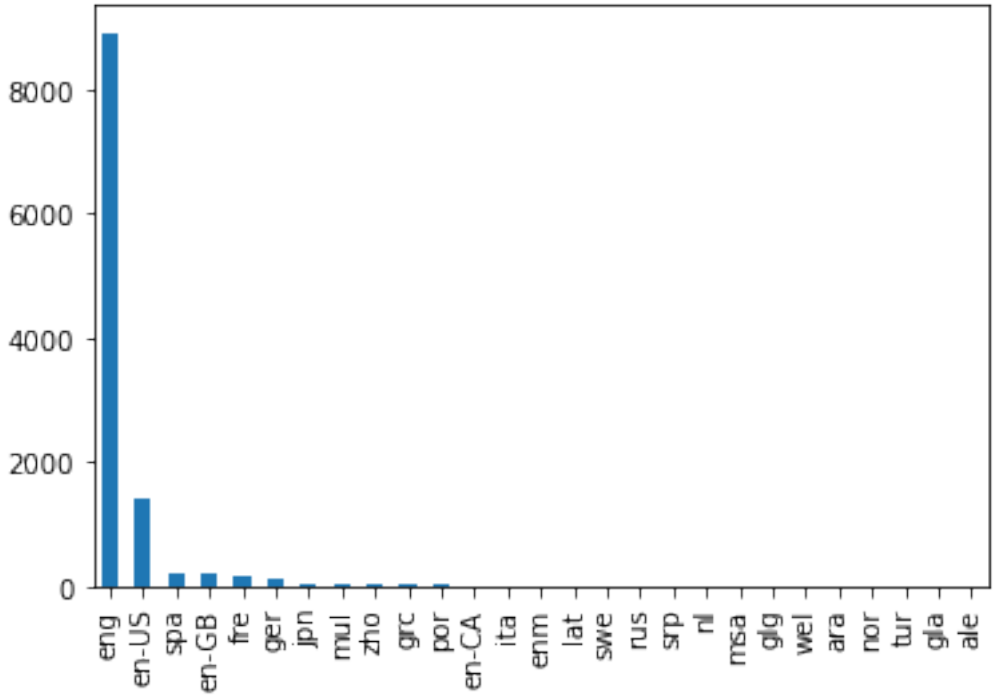
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From␣
 ↪version
0.12, the only valid positional argument will be `data`, and passing␣
 ↪other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[13]:
```
Text(0.5, 0, 'Total Ratings')
```



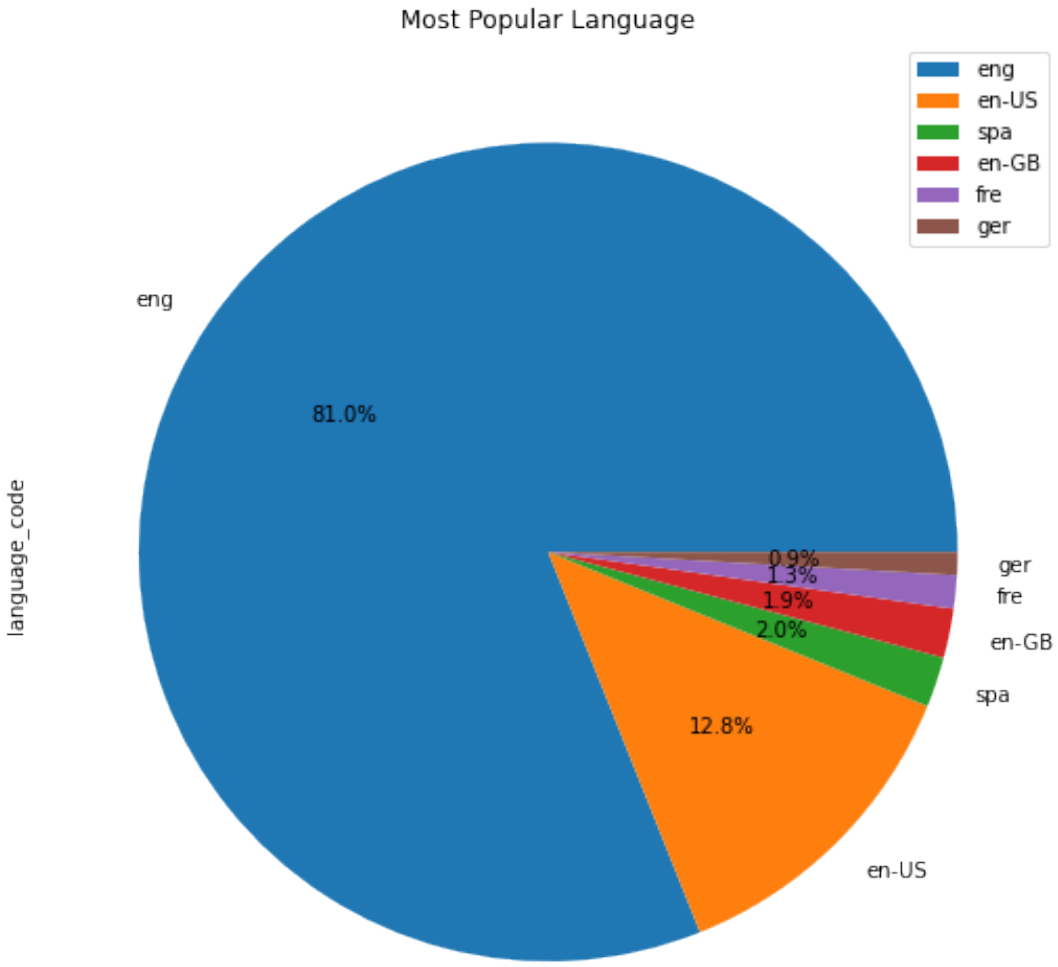
[14]:
```
# top languages
data['language_code'].value_counts().plot(kind='bar')
```

[14]: <AxesSubplot:>



[15]:
```
plt.title('Most Popular Language')
plt.ylabel('Counts')
plt.xticks(rotation = 90)
data['language_code'].value_counts().head(6).plot(kind = 'pie',␣
 ↪autopct='%1.1f%%', figsize=(9, 9)).legend()
```

[15]: <matplotlib.legend.Legend at 0x1dcd1d5f160>



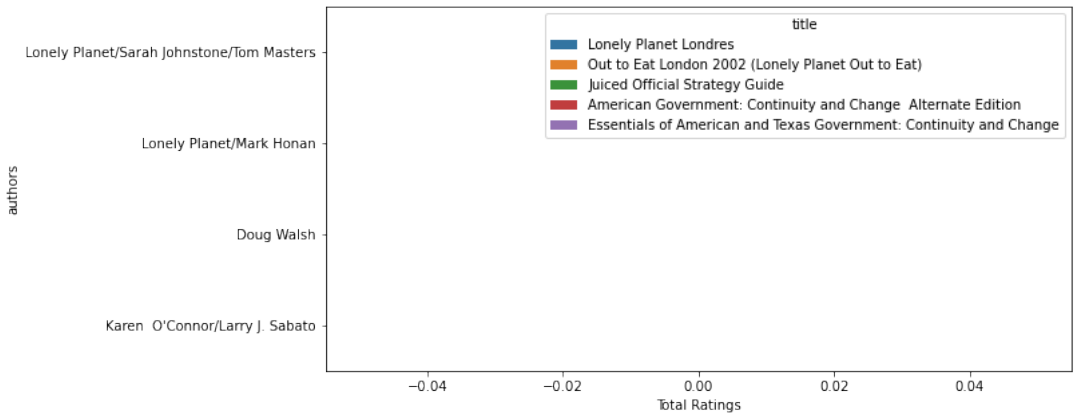
We can see that the dominant language is English, with a much higher percentage than other languages, this being: 95.7%. Therefore, we can see that the authors and titles with the best ranking are in this language, since it is the most used to produce books.

[16]:
```
# authors with smallets rated books
plt.figure(figsize=(10, 5))
authors = data.nsmallest(5, ['ratings_count']).set_index('authors')
sns.barplot(authors['ratings_count'], authors.index, ci = None, hue =␣
 ↪authors['title'])
plt.xlabel('Total Ratings')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From␣
 ↪version
```

```
0.12, the only valid positional argument will be `data`, and passing␣
 ↪other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

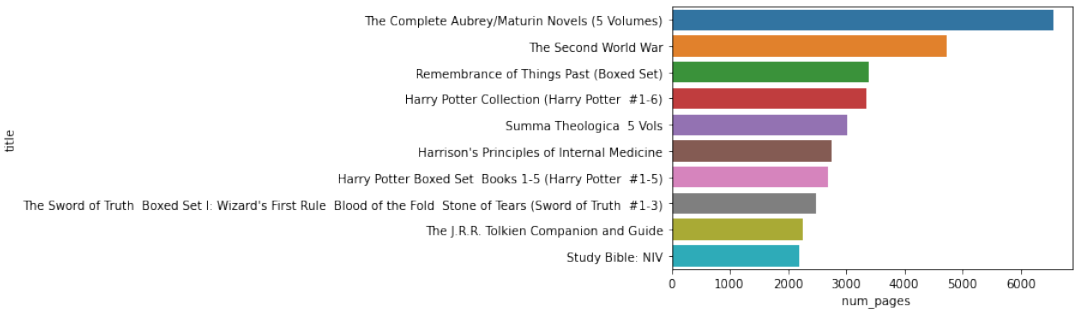
[16]:
```
Text(0.5, 0, 'Total Ratings')
```



[17]:
```
# top 10 longest books
longest_books = data.nlargest(10, ['  num_pages']).set_index('title')
sns.barplot(longest_books['  num_pages'], longest_books.index)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From␣
 ↪version
0.12, the only valid positional argument will be `data`, and passing␣
 ↪other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

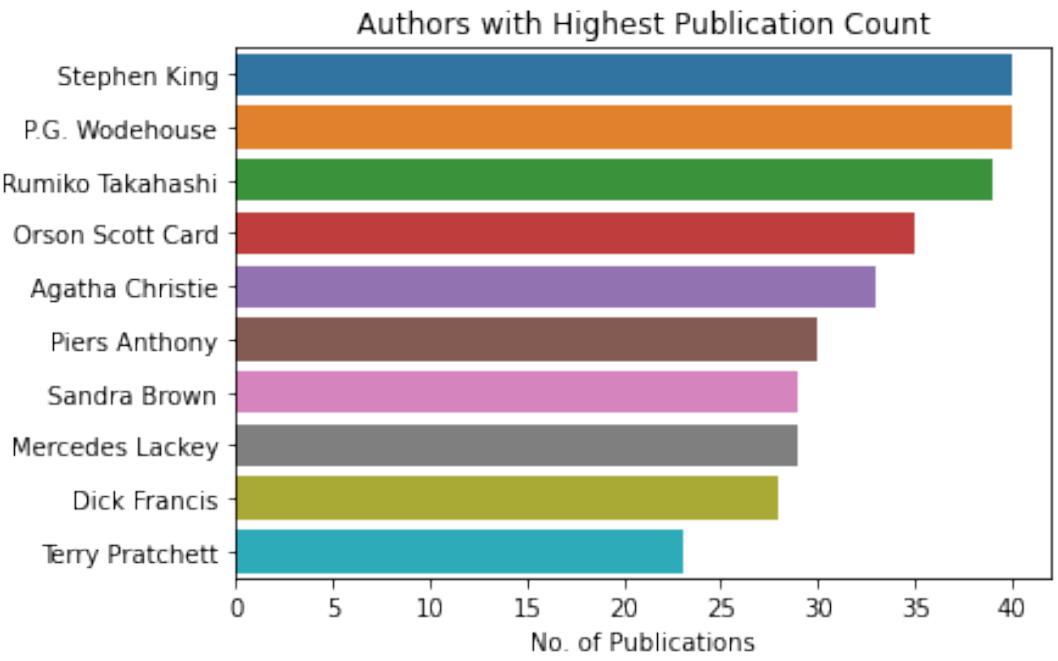
[17]:
```
<AxesSubplot:xlabel='  num_pages', ylabel='title'>
```



[18]:
```
# authors with highest publications
top_authors = data['authors'].value_counts().head(10)
sns.barplot(top_authors, top_authors.index)
plt.title('Authors with Highest Publication Count')
plt.xlabel('No. of Publications')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From␣
 ↪version
0.12, the only valid positional argument will be `data`, and passing␣
 ↪other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[18]:
```
Text(0.5, 0, 'No. of Publications')
```

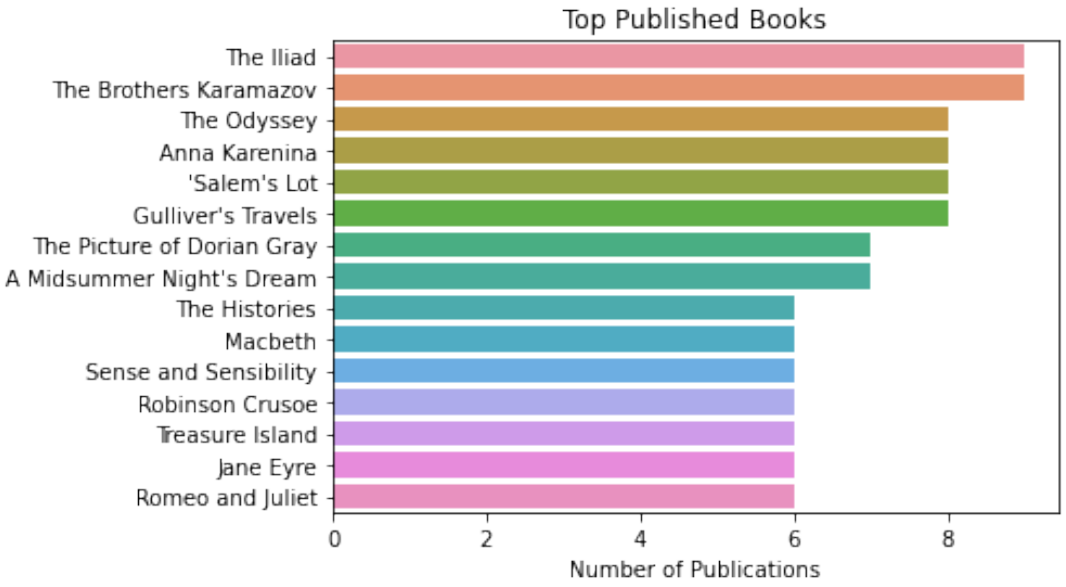


[19]:
```
# top published books
sns.barplot(data['title'].value_counts()[:15], data['title'].
 ↪value_counts().index[:15])
plt.title('Top Published Books')
plt.xlabel('Number of Publications')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From␣
 ↪version
```

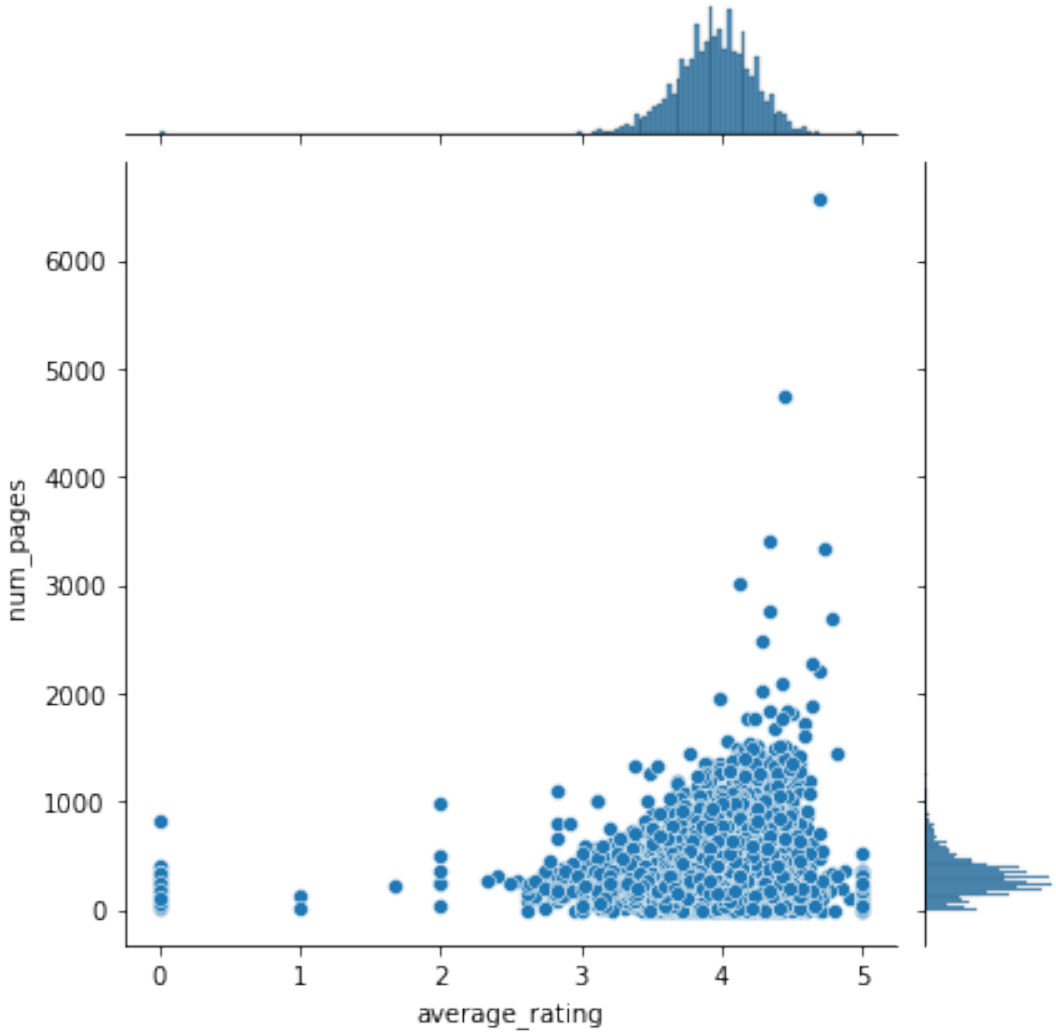
```
0.12, the only valid positional argument will be `data`, and passing␣
 ↪other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[19]:
```
Text(0.5, 0, 'Number of Publications')
```



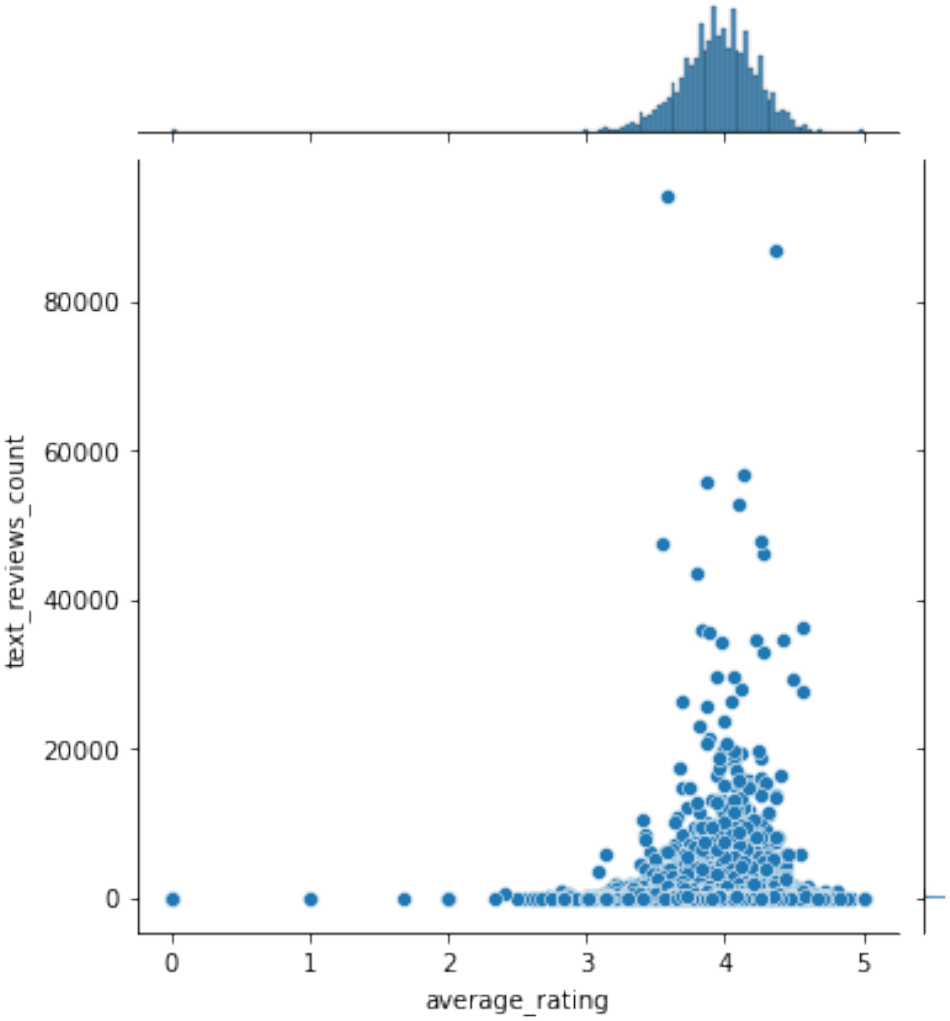
[20]:
```
# visualise a bivariate distribution between ratings & no. of pages
sns.jointplot(x = 'average_rating', y = '  num_pages', data = data)
```

[20]:
```
<seaborn.axisgrid.JointGrid at 0x1dcd304cca0>
```



[21]:
```
# visualise a bivariate distribution between ratings & no. of reviews
sns.jointplot(x = 'average_rating', y = 'text_reviews_count', data =␣
 ↪data)
```

[21]: <seaborn.axisgrid.JointGrid at 0x1dcd3610eb0>

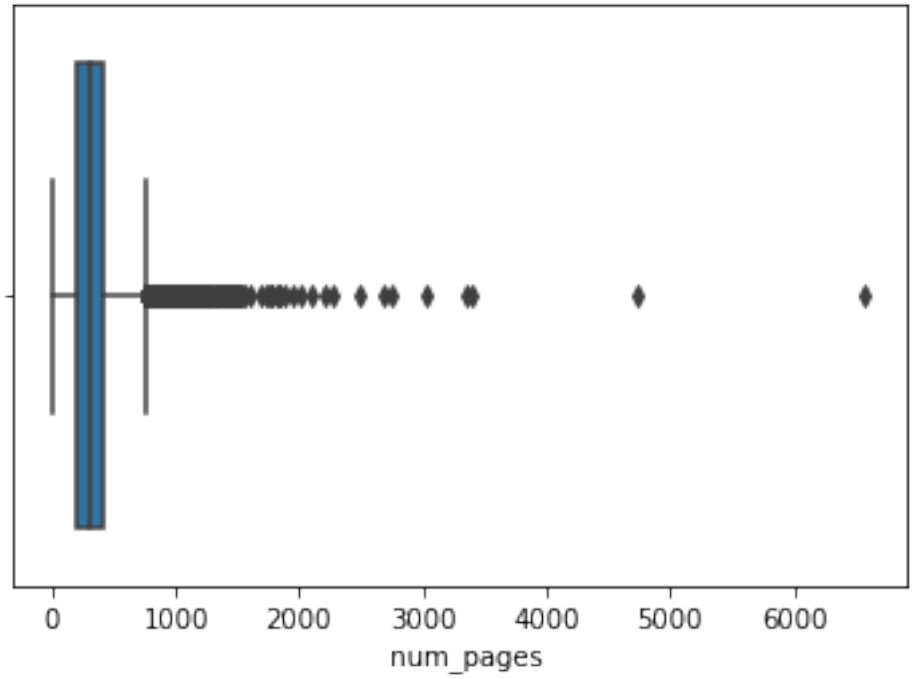


ML model and data processing version 1

Data preprocessing

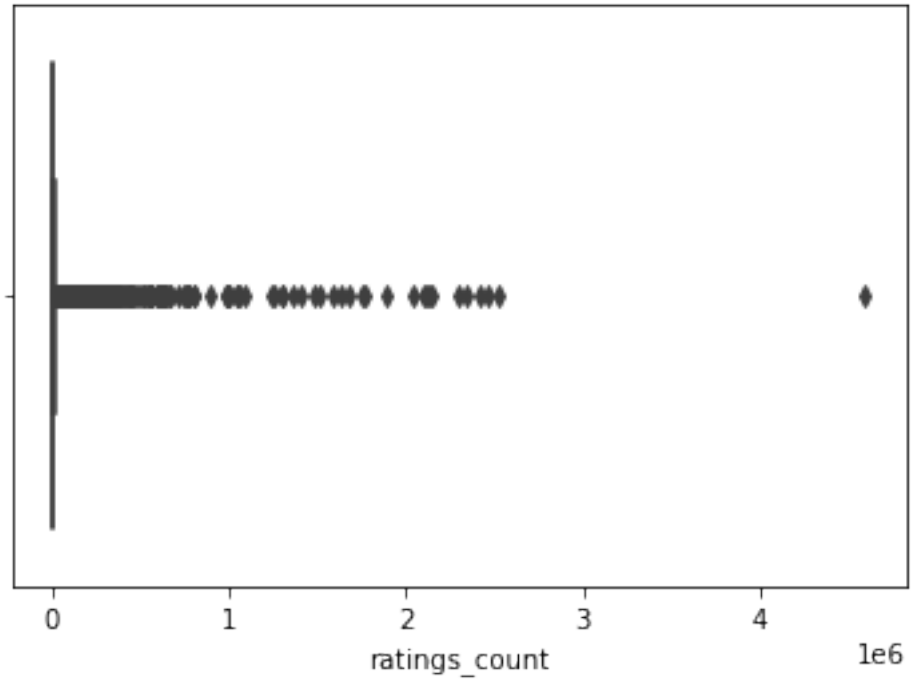
[22]:
```
# find no. of pages outliers
sns.boxplot(x=data['  num_pages'])
```

[22]:
```
<AxesSubplot:xlabel='  num_pages'>
```



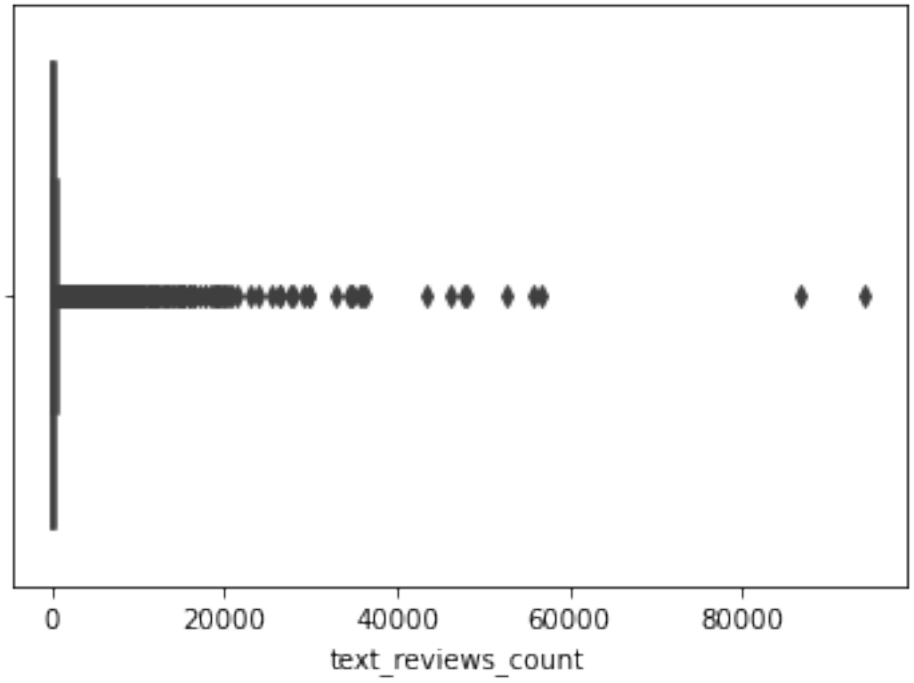
[23]:
```
# find ratings count outliers
sns.boxplot(x=data['ratings_count'])
```

[23]: <AxesSubplot:xlabel='ratings_count'>



[24]:
```
# find ratings count outliers
sns.boxplot(x=data['text_reviews_count'])
```

[24]: <AxesSubplot:xlabel='text_reviews_count'>



Feature Engineering

[25]:
```
# encode title column
le = preprocessing.LabelEncoder()
data['title'] = le.fit_transform(data['title'])
```

[26]:
```
# encode authors column
data['authors'] = le.fit_transform(data['authors'])
```

[27]:
```
# encode language column
enc_lang = pd.get_dummies(data['language_code'])
data = pd.concat([data, enc_lang], axis = 1)
```

[28]:
```
#encode publisher
data['publisher'] = le.fit_transform(data['publisher'])
```

[29]:
```
#encode publication_date
data['publication_date'] = le.fit_transform(data['publication_date'])
```

*Machine Learning Model

[30]:
```
# divide the data into attributes and labels
X = data.drop(['average_rating', 'language_code', 'isbn', 'isbn13'],␣
 ↪axis = 1)
```

[31]:
```
y = data['average_rating']
```

[32]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.
 ↪2, random_state = 0)
```

[33]:
```
'X_train shape: ', X_train.shape
```

[33]:
```
('X_train shape: ', (8898, 35))
```

[34]:
```
'X_test shape: ',X_test.shape
```

[34]:
```
('X_test shape: ', (2225, 35))
```

[35]:
```
'y_test shape: ',y_test.shape
```

[35]:
```
('y_test shape: ', (2225,))
```

[36]:
```
'y_train shape: ',y_train.shape
```

[36]:
```
('y_train shape: ', (8898,))
```

[37]:
```
lr = LinearRegression()
```

[38]:
```
lr.fit(X_train, y_train)
```

[38]:
```
LinearRegression()
```

[39]:
```
predictions = lr.predict(X_test)
```

[40]:
```
predictions
```

[40]:
```
array([4.0622979 , 3.96285018, 3.9758366 , ..., 3.94102896, 3.90644686,
       3.97393885])
```

[41]:
```
pred = pd.DataFrame({'Actual': y_test.tolist(), 'Predicted':␣
 ↪predictions.tolist()}).head(25)
```

[42]:
```
pred.head(10)
```

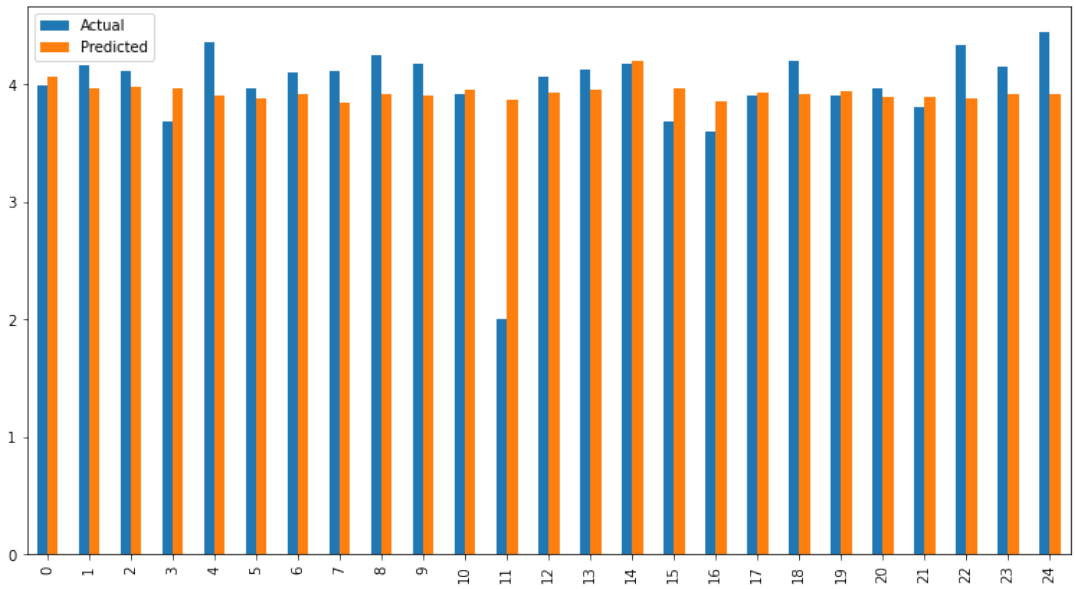
[42]:

	Actual	Predicted
0	3.99	4.062298
1	4.16	3.962850
2	4.12	3.975837
3	3.68	3.963737

```
4     4.36    3.904451
5     3.97    3.878351
6     4.10    3.921220
7     4.12    3.845463
8     4.25    3.922268
9     4.17    3.904506
```

[43]:
```
# visualise the above comparison result
pred.plot(kind='bar', figsize=(13, 7))
```

[43]: <AxesSubplot:>



[44]:
```
print('MAE:', metrics.mean_absolute_error(y_test, predictions))
print('MSE:', metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
```

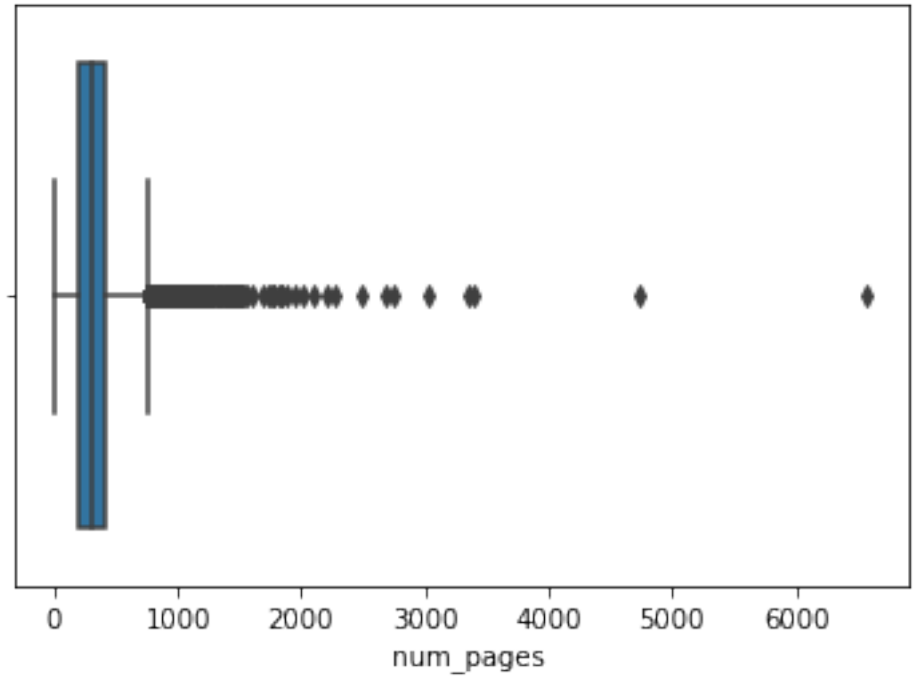
```
MAE: 0.22785560782959874
MSE: 0.11598507711983519
RMSE: 0.34056581907149047
```

ML model and data processing version 2

Data preprocessing

[22]:
```
# find no. of pages outliers
sns.boxplot(x=data['  num_pages'])
```

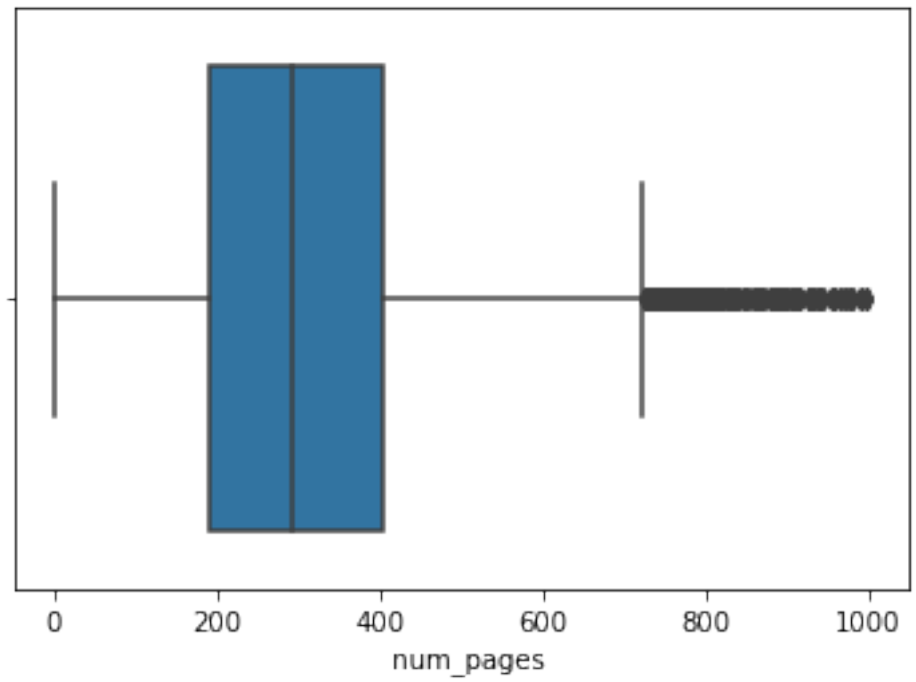
[22]: <AxesSubplot:xlabel=' num_pages'>



[23]:
```
# remove outliers from no. of pages
data = data.drop(data.index[data['  num_pages'] >= 1000])
```

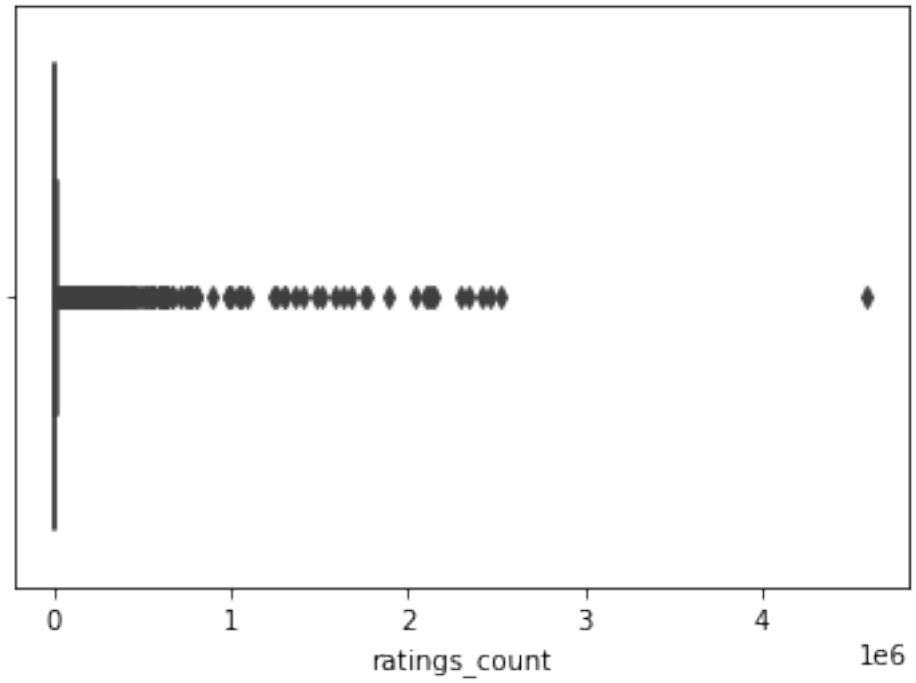
[24]:
```
sns.boxplot(x=data['  num_pages'])
```

[24]: <AxesSubplot:xlabel=' num_pages'>



[25]:
```
# find ratings count outliers
sns.boxplot(x=data['ratings_count'])
```

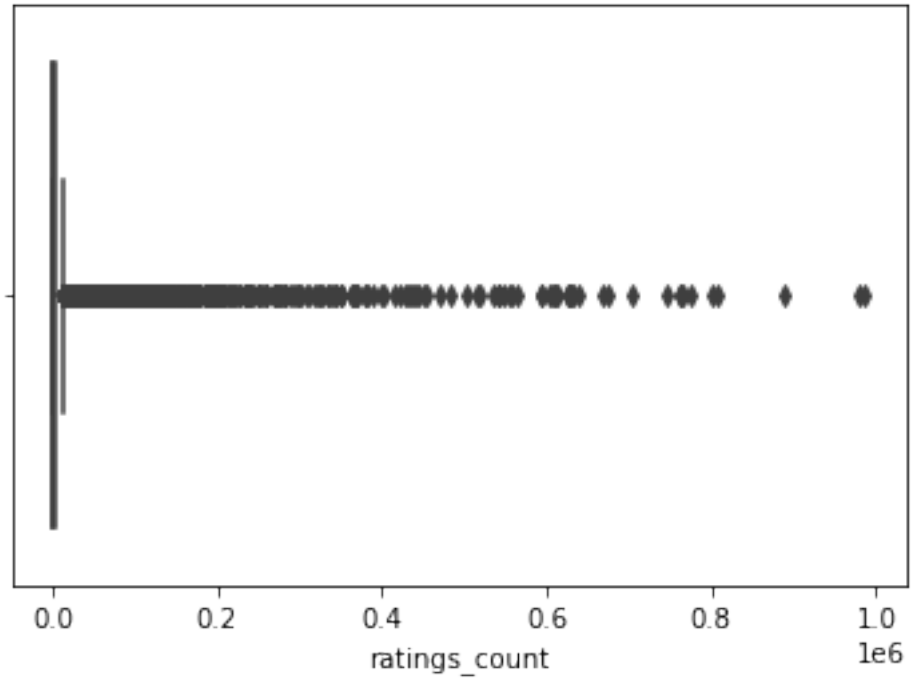
[25]: <AxesSubplot:xlabel='ratings_count'>



[26]:
```
# remove outliers from ratings_count
data = data.drop(data.index[data['ratings_count'] >= 1000000])
```

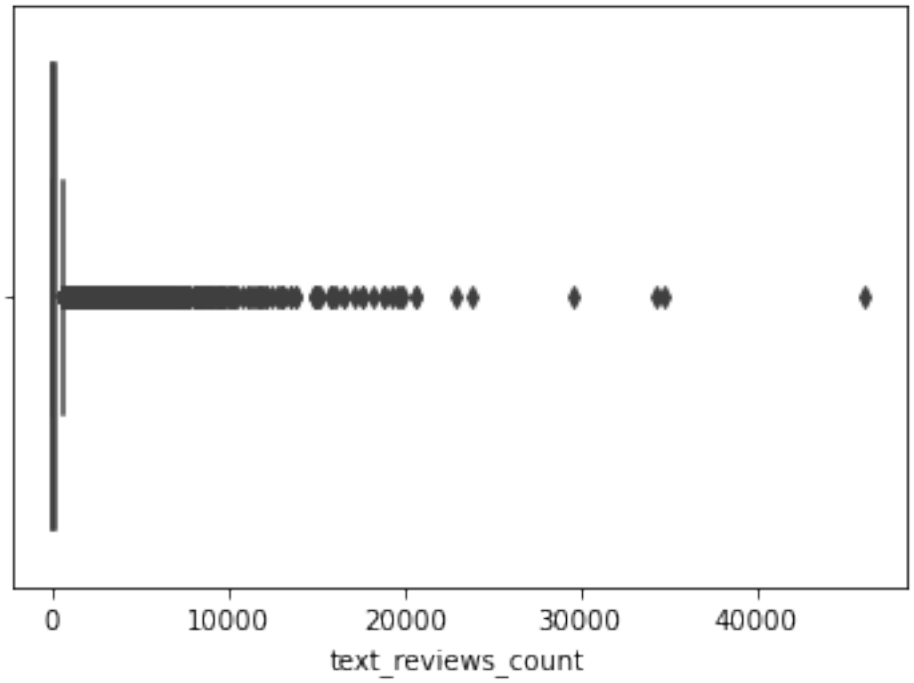
[27]:
```
sns.boxplot(x=data['ratings_count'])
```

[27]: <AxesSubplot:xlabel='ratings_count'>



[28]:
```
# find ratings count outliers
sns.boxplot(x=data['text_reviews_count'])
```

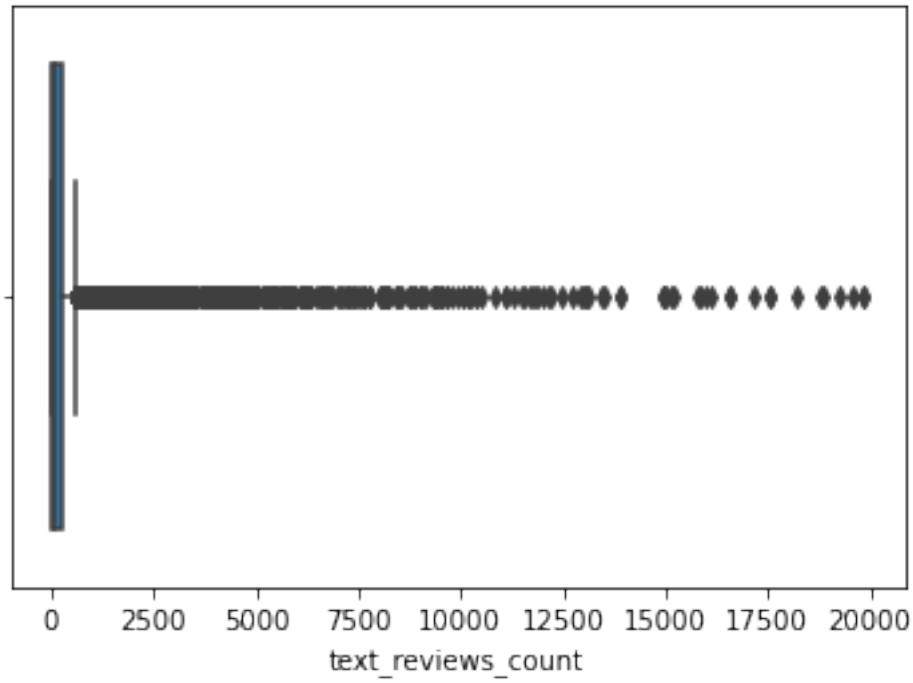
[28]: <AxesSubplot:xlabel='text_reviews_count'>



[29]:
```
# remove outliers from text_reviews_count
data = data.drop(data.index[data['text_reviews_count'] >= 20000])
```

[30]:
```
sns.boxplot(x=data['text_reviews_count'])
```

[30]:
```
<AxesSubplot:xlabel='text_reviews_count'>
```



Feature Engineering

[31]:
```
# encode title column
le = preprocessing.LabelEncoder()
```

[32]:
```
data['title'] = le.fit_transform(data['title'])
```

[33]:
```
# encode authors column
data['authors'] = le.fit_transform(data['authors'])
```

[34]:
```
# encode language column
enc_lang = pd.get_dummies(data['language_code'])
data = pd.concat([data, enc_lang], axis = 1)
```

[35]:
```
#encode publisher
data['publisher'] = le.fit_transform(data['publisher'])
```

[36]:
```
#encode publication_date
data['publication_date'] = le.fit_transform(data['publication_date'])
```

Machine Learning Model

[37]:
```
# divide the data into attributes and labels
X = data.drop(['average_rating', 'language_code', 'isbn', 'isbn13'],␣
 ↪axis = 1)
```

[38]:
```
y = data['average_rating']
```

[39]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.
 ↪2, random_state = 0)
```

[40]:
```
'X_train: ', X_train.shape
```

[40]:
```
('X_train: ', (8694, 35))
```

[41]:
```
'X_test: ',X_test.shape
```

[41]:
```
('X_test: ', (2174, 35))
```

[42]:
```
'y_train: ',y_train.shape
```

[42]:
```
('y_train: ', (8694,))
```

[43]:
```
'y_test: ',y_test.shape
```

[43]:
```
('y_test: ', (2174,))
```

[44]:
```
lr = LinearRegression()
```

[45]:
```
lr.fit(X_train, y_train)
```

[45]:
```
LinearRegression()
```

[46]:
```
predictions = lr.predict(X_test)
```

[47]:
```
pred = pd.DataFrame({'Actual': y_test.tolist(), 'Predicted':␣
 ↪predictions.tolist()}).head(25)
```

[48]:
```
pred.head(10)
```

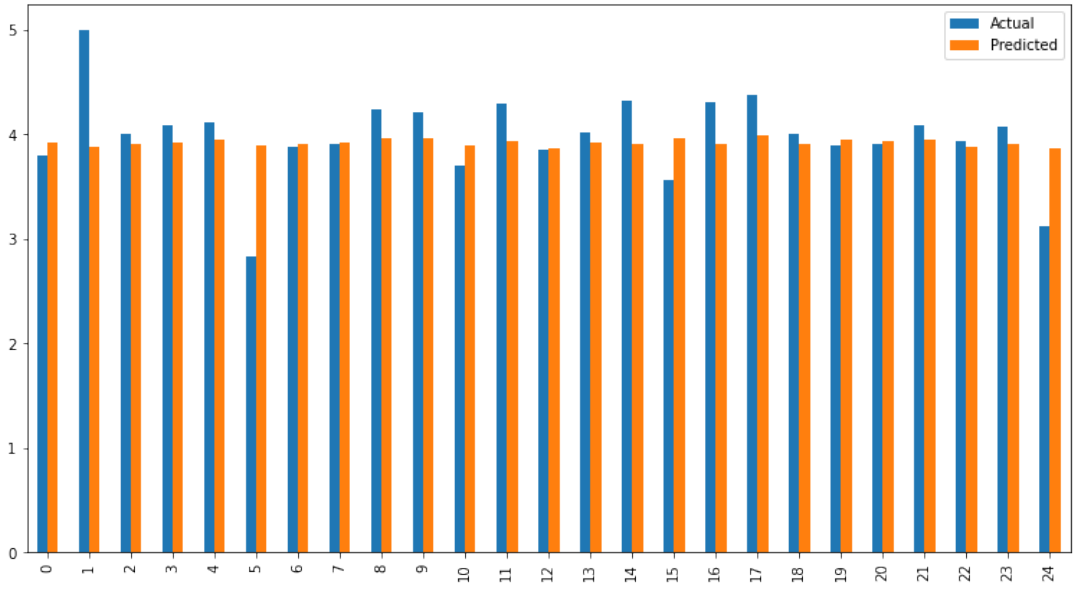
[48]:

	Actual	Predicted
0	3.80	3.912720
1	5.00	3.874783
2	4.00	3.901000

```
3    4.09   3.924655
4    4.11   3.944028
5    2.83   3.887576
6    3.88   3.908823
7    3.91   3.918955
8    4.23   3.962275
9    4.21   3.958037
```

[49]:
```
# visualise the above comparison result
pred.plot(kind='bar', figsize=(13, 7))
```

[49]: <AxesSubplot:>



[50]:
```
print('MAE:', metrics.mean_absolute_error(y_test, predictions))
print('MSE:', metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
```

```
MAE: 0.22401505196756463
MSE: 0.10527048686775198
RMSE: 0.32445413677090323
```

[22]:
```
ML model and data processing version 3
```

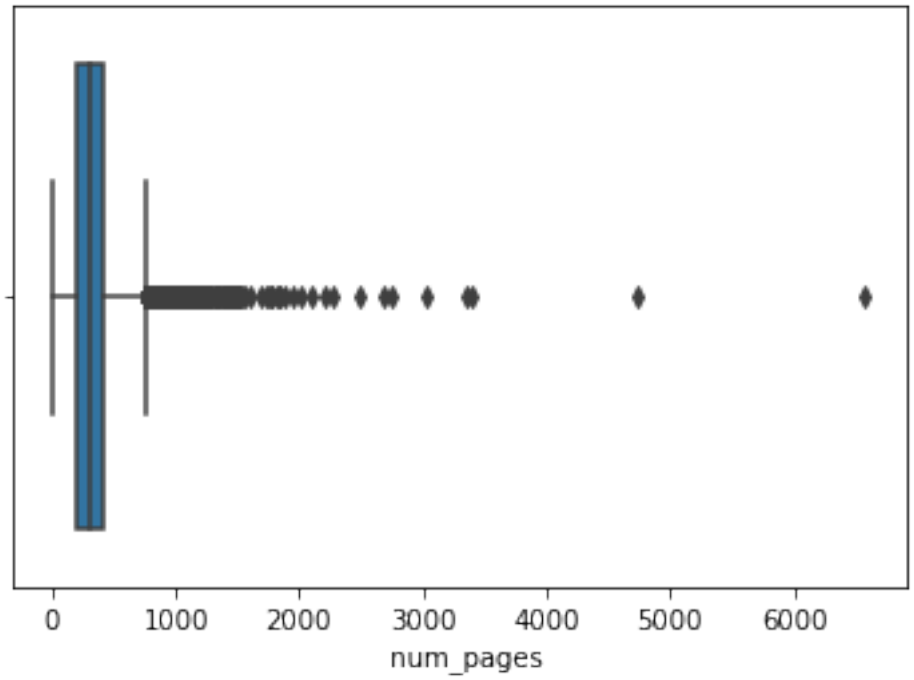
```
  File "C:\Users\gmgar\AppData\Local\Temp/ipykernel_9640/1909486236.py"
↪line 1
    ML model and data processing version 3
       ^
SyntaxError: invalid syntax
```

ML model and data processing version 3

Data preprocessing

[23]:
```
# find no. of pages outliers
sns.boxplot(x=data['  num_pages'])
```

[23]:
```
<AxesSubplot:xlabel='  num_pages'>
```



[24]:
```
np.percentile(data['  num_pages'],[99])
```

[24]:
```
array([1174.24])
```

[25]:
```
np.percentile(data['  num_pages'],[99])[0]
```

[25]:
```
1174.2400000000052
```

[26]:
```
uv = np.percentile(data['  num_pages'],[99])[0]
```

[27]:
```
datos_mayores_que_uv =data[data['  num_pages'] > uv]
```

[28]:
```
datos_mayores_que_uv
```

[28]:

	bookID	title	authors	average_rating	isbn	isbn13	language_code	num_pages	ratings_count
4	8	Harry Potter Boxed Set Books 1-5 (Harry Potte...	J.K. Rowling/Mary GrandPré	4.78	0439682584	9780439682589	eng	2690	41428
6	10	Harry Potter Collection (Harry Potter #1-6)	J.K. Rowling	4.73	0439827604	9780439827607	eng	3342	28242
21	30	J.R.R. Tolkien 4-Book Boxed Set: The Hobbit an...	J.R.R. Tolkien	4.59	0345538374	9780345538376	eng	1728	101233
22	31	The Lord of the Rings (The Lord of the Rings ...	J.R.R. Tolkien	4.50	0618517650	9780618517657	eng	1184	1710
24	35	The Lord of the Rings (The Lord of the Rings ...	J.R.R. Tolkien/Alan Lee	4.50	0618260587	9780618260584	en-US	1216	1618
...	...	...	...	...	...	...	...	...	...
10378	42054	The Arden Shakespeare Complete Works	William Shakespeare/Richard Proudfoot/Ann Thom...	4.50	1903436613	9781903436615	eng	1347	122
10534	42929	Gai-Jin (Asian Saga #3)	James Clavell	3.86	044021680X	9780440216803	eng	1236	10757
10535	42932	Whirlwind (Asian Saga #6)	James Clavell	3.82	0340766182	9780340766187	eng	1231	5626
10749	43888	The Sword of Truth Boxed Set I: Wizard's Firs...	Terry Goodkind	4.29					
10906	44613	Remembrance of Things Past (Boxed Set)	Marcel Proust/C.K. Scott Moncrieff/Frederick A...	4.34					

```
10749  0812575601  9780812575606         en-US         2480          ␣
↪4196
10906  0701125594  9780701125592           eng         3400          ␣
↪ 6
```

	text_reviews_count	publication_date	publisher
4	164	9/13/2004	Scholastic
6	808	9/12/2005	Scholastic
21	1550	9/25/2012	Ballantine Books
22	91	10/21/2004	Houghton Mifflin Harcourt
24	140	10/1/2002	Houghton Mifflin Harcourt
...	...	...	...
10378	7	7/5/2001	Arden Shakespeare
10534	296	4/3/1994	Dell
10535	164	12/2/1999	Morrow
10749	81	11/15/1998	Tor Books
10906	1	3/5/1981	Chatto & Windus

[112 rows x 12 columns]

[29]:
```
data['  num_pages'][(data['  num_pages']> 3*uv)] = 3*uv
```

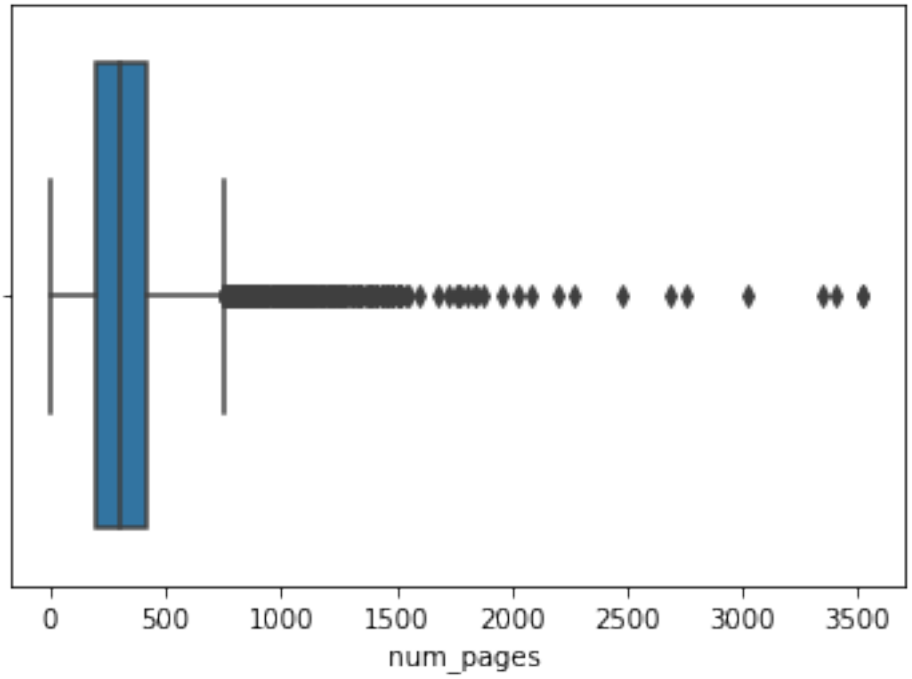
```
C:\Users\gmgar\AppData\Local\Temp/ipykernel_9640/16307480.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['  num_pages'][(data['  num_pages']> 3*uv)] = 3*uv
```

[30]:
```
datos_mayores_que_uv =data[data['  num_pages'] > uv]
```

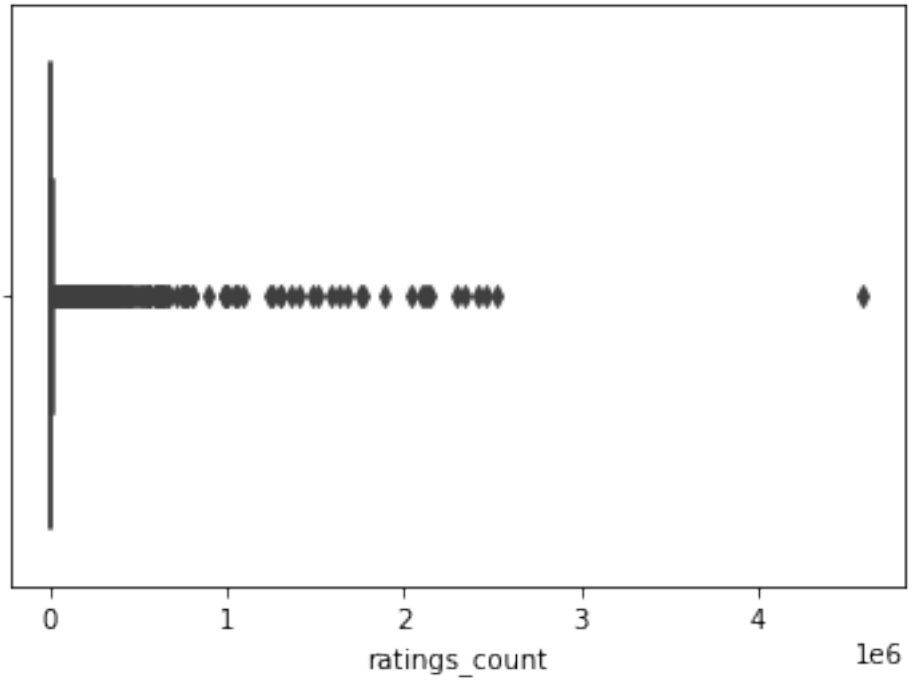
[31]:
```
sns.boxplot(x=data['  num_pages'])
```

[31]:
```
<AxesSubplot:xlabel='  num_pages'>
```



[32]:
```
# find ratings count outliers
sns.boxplot(x=data['ratings_count'])
```

[32]: <AxesSubplot:xlabel='ratings_count'>



[33]:
```
np.percentile(data['ratings_count'],[99])
```

[33]:
```
array([297462.16])
```

[34]:
```
np.percentile(data['ratings_count'],[99])[0]
```

[34]:
```
297462.1600000002
```

[35]:
```
uw = np.percentile(data['ratings_count'],[99])[0]
```

[36]:
```
datos_mayores_que_uw =data[data['ratings_count'] > uw]
```

[37]:
```
datos_mayores_que_uw
```

[37]:

	bookID	title	authors	average_rating	isbn	isbn13
0	1	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPré	4.57	0439785960	9780439785969
1	2	Harry Potter and the Order of the Phoenix (Har...	J.K. Rowling/Mary GrandPré	4.49	0439358078	9780439358071
3	5	Harry Potter and the Prisoner of Azkaban (Harr...	J.K. Rowling/Mary GrandPré	4.56	043965548X	9780439655484
23	34	The Fellowship of the Ring (The Lord of the Ri...	J.R.R. Tolkien	4.36	0618346252	9780618346257
139	295	Treasure Island	Robert Louis Stevenson	3.83	0753453800	9780753453803
...	...	...	...	...	...	...
9951	40102	Blink: The Power of Thinking Without Thinking	Malcolm Gladwell	3.93	0316010669	9780316010665
10336	41865	Twilight (Twilight #1)	Stephenie Meyer	3.59	0316015849	9780316015844
10395	42156	Something Borrowed (Darcy & Rachel #1)	Emily Giffin	3.85	031232118X	9780312321185
10700	43641	Water for Elephants				
10728	43763	Interview with the Vampire (The Vampire Chroni...				

```
10700                  Sara Gruen            4.09  1565125606 ␣
↪9781565125605
10728                   Anne Rice            3.99  0345476875 ␣
↪9780345476876

      language_code    num_pages  ratings_count  text_reviews_count  \
0               eng        652.0        2095690               27591
1               eng        870.0        2153167               29221
3               eng        435.0        2339585               36325
23              eng        398.0        2128944               13670
139             eng        311.0         318753                6734
...             ...          ...            ...                 ...
9951            eng        296.0         437507               12937
10336           eng        501.0        4597666               94265
10395           eng        322.0         454917                7487
10700           eng        335.0        1260027               52759
10728           eng        342.0         433413                7368

      publication_date                publisher
0            9/16/2006            Scholastic Inc.
1             9/1/2004            Scholastic Inc.
3             5/1/2004            Scholastic Inc.
23            9/5/2003  Houghton Mifflin Harcourt
139           9/15/2001                Kingfisher
...                 ...                       ...
9951          4/3/2007             Back Bay Books
10336         9/6/2006  Little  Brown and Company
10395         6/1/2004        St. Martin's Press
10700         5/1/2007           Algonquin Books
10728        8/31/2004          Ballantine Books

[112 rows x 12 columns]
```

[38]:
```
data['ratings_count'][(data['ratings_count']> 3*uw)] = 3*uw
```

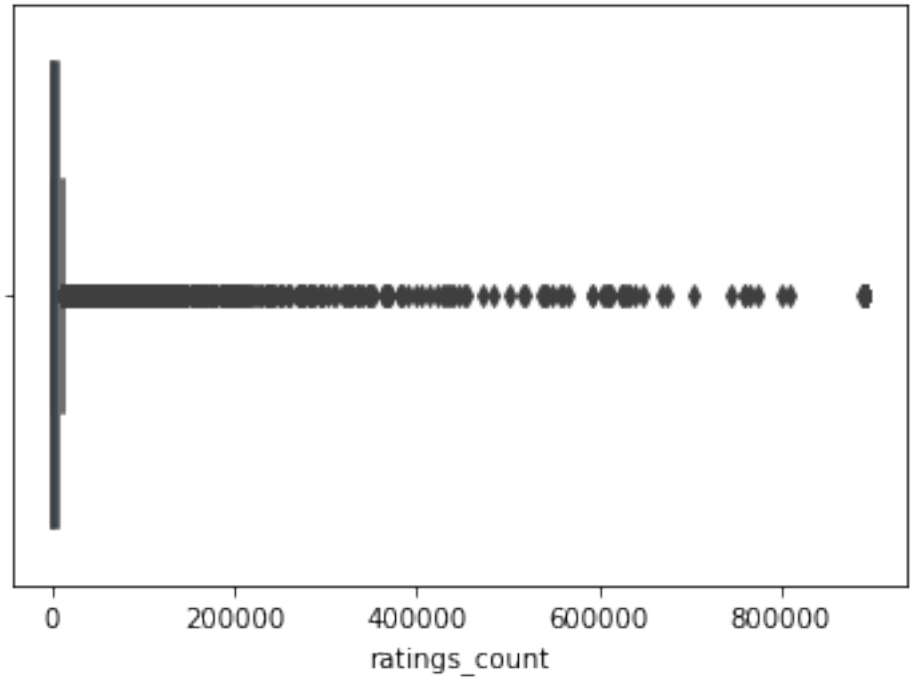
```
C:\Users\gmgar\AppData\Local\Temp/ipykernel_9640/336942012.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['ratings_count'][(data['ratings_count']> 3*uw)] = 3*uw
```

[39]:
```
datos_mayores_que_uw =data[data['ratings_count'] > uw]
```

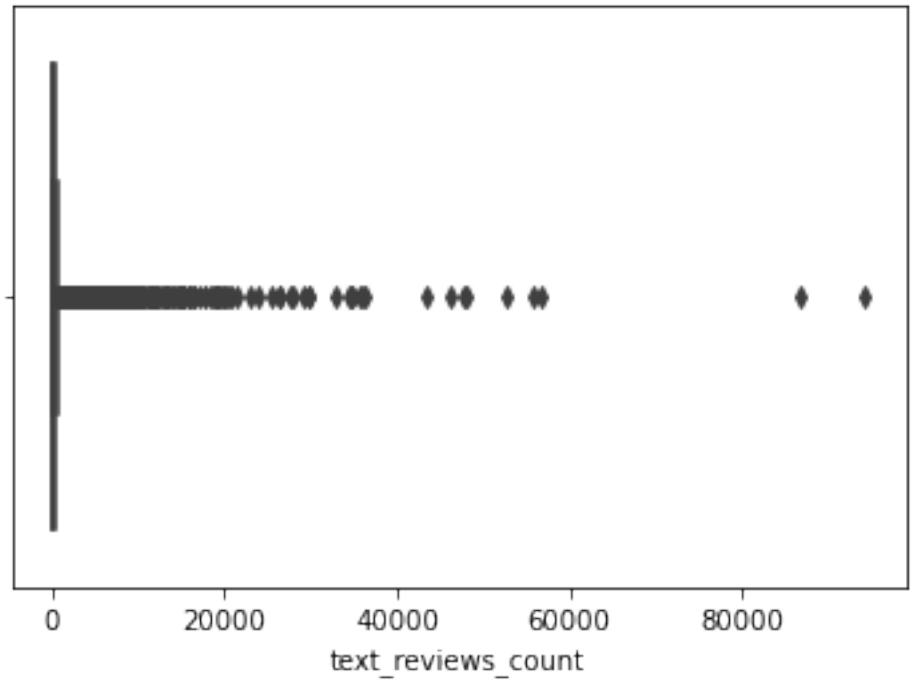
[40]:
```
sns.boxplot(x=data['ratings_count'])
```

[40]:
```
<AxesSubplot:xlabel='ratings_count'>
```



[41]:
```
# find ratings count outliers
sns.boxplot(x=data['text_reviews_count'])
```

[41]: <AxesSubplot:xlabel='text_reviews_count'>



[42]:
```
np.percentile(data['text_reviews_count'],[99])
```

[42]:
```
array([8978.92])
```

[43]:
```
np.percentile(data['text_reviews_count'],[99])[0]
```

[43]:
```
8978.920000000075
```

[44]:
```
ux = np.percentile(data['text_reviews_count'],[99])[0]
```

[45]:
```
datos_mayores_que_ux =data[data['text_reviews_count'] > ux]
```

[46]:
```
datos_mayores_que_ux
```

[46]:

	bookID	title	authors	average_rating	isbn	isbn13
0	1	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPré	4.57	0439785960	9780439785969
1	2	Harry Potter and the Order of the Phoenix (Har...	J.K. Rowling/Mary GrandPré	4.49	0439358078	9780439358071
3	5	Harry Potter and the Prisoner of Azkaban (Harr...	J.K. Rowling/Mary GrandPré	4.56	043965548X	9780439655484
12	21	A Short History of Nearly Everything	Bill Bryson	4.21	076790818X	9780767908184
23	34	The Fellowship of the Ring (The Lord of the Ri...	J.R.R. Tolkien	4.36	0618346252	9780618346257
...	...	...	...	...	...	...
10039	40440	The Thirteenth Tale	Diane Setterfield	3.96	0743298020	9780743298025
10336	41865	Twilight (Twilight #1)	Stephenie Meyer	3.59	0316015849	9780316015844
10527	42899	Dark Lover (Black Dagger Brotherhood #1)	J.R. Ward	4.20	0451216954	9780451216953
10549	43015	A Long Way Gone: Memoirs of a Boy Soldier				
10700	43641	Water for Elephants				

```
10549                Ishmael Beah            4.16  0374105235 ␣
↪9780374105235
10700                  Sara Gruen            4.09  1565125606 ␣
↪9781565125605

      language_code    num_pages  ratings_count  text_reviews_count  \
0               eng        652.0      892386.48               27591
1               eng        870.0      892386.48               29221
3               eng        435.0      892386.48               36325
12              eng        544.0      248558.00                9396
23              eng        398.0      892386.48               13670
...             ...          ...            ...                 ...
10039           eng        406.0      239809.00               18865
10336           eng        501.0      892386.48               94265
10527           eng        393.0      259511.00               10475
10549           eng        229.0      147820.00                9547
10700           eng        335.0      892386.48               52759

      publication_date                 publisher
0            9/16/2006           Scholastic Inc.
1             9/1/2004           Scholastic Inc.
3             5/1/2004           Scholastic Inc.
12           9/14/2004            Broadway Books
23            9/5/2003  Houghton Mifflin Harcourt
...                ...                       ...
10039        9/12/2006              Atria Books
10336         9/6/2006  Little  Brown and Company
10527         9/6/2005       Penguin Group (USA)
10549        2/13/2007      Sarah Crichton Books
10700         5/1/2007           Algonquin Books

[112 rows x 12 columns]
```

[47]:
```
data['text_reviews_count'][(data['text_reviews_count']> 3*ux)] = 3*ux
```

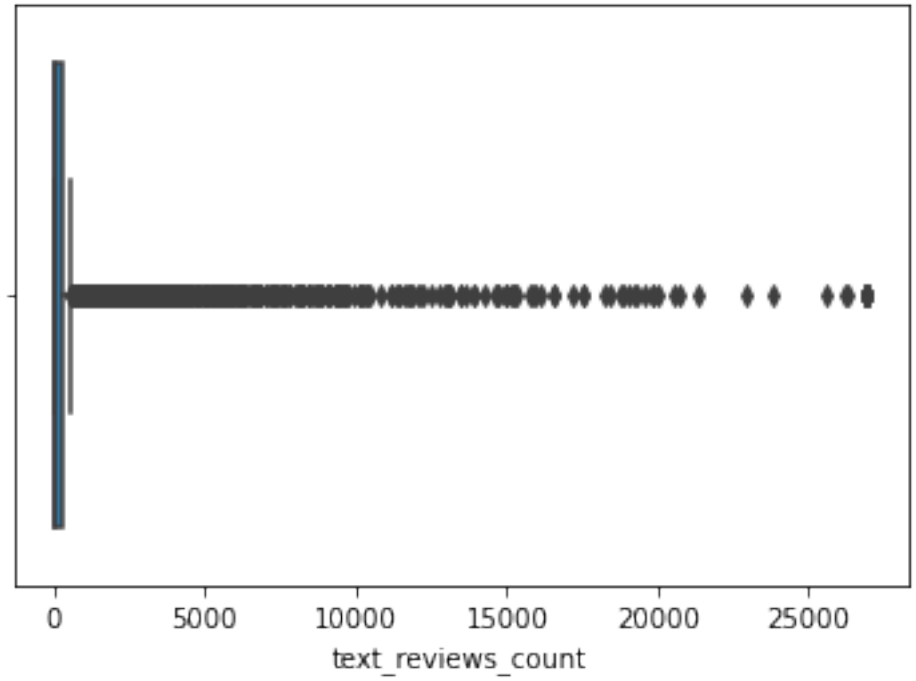
```
C:\Users\gmgar\AppData\Local\Temp/ipykernel_9640/3041484283.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['text_reviews_count'][(data['text_reviews_count']> 3*ux)] = 3*ux
```

[48]:
```
datos_mayores_que_ux =data[data['text_reviews_count'] > ux]
```

[49]:
```
sns.boxplot(x=data['text_reviews_count'])
```

[49]:
```
<AxesSubplot:xlabel='text_reviews_count'>
```



Feature Engineering

[50]:
```
# encode title column
le = preprocessing.LabelEncoder()
data['title'] = le.fit_transform(data['title'])
```

[51]:
```
# encode authors column
data['authors'] = le.fit_transform(data['authors'])
```

[52]:
```
# encode language column
enc_lang = pd.get_dummies(data['language_code'])
data = pd.concat([data, enc_lang], axis = 1)
```

[53]:
```
#encode publisher
data['publisher'] = le.fit_transform(data['publisher'])
```

[54]:
```
#encode publication_date
data['publication_date'] = le.fit_transform(data['publication_date'])
```

Machine Learning Model

[55]:
```
# divide the data into attributes and labels
X = data.drop(['average_rating', 'language_code', 'isbn', 'isbn13'],␣
 ↪axis = 1)
```

[56]:
```
y = data['average_rating']
```

[57]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.
↪2, random_state = 0)
```

[58]:
```
'X_train: ', X_train.shape
```

[58]:
```
('X_train: ', (8898, 35))
```

[59]:
```
'X_test: ',X_test.shape
```

[59]:
```
('X_test: ', (2225, 35))
```

[60]:
```
'y_train: ',y_train.shape
```

[60]:
```
('y_train: ', (8898,))
```

[61]:
```
'y_test: ',y_test.shape
```

[61]:
```
('y_test: ', (2225,))
```

[62]:
```
lr = LinearRegression()
```

[63]:
```
lr.fit(X_train, y_train)
```

[63]:
```
LinearRegression()
```

[64]:
```
predictions = lr.predict(X_test)
```

[65]:
```
pred = pd.DataFrame({'Actual': y_test.tolist(), 'Predicted':␣
↪predictions.tolist()}).head(25)
```

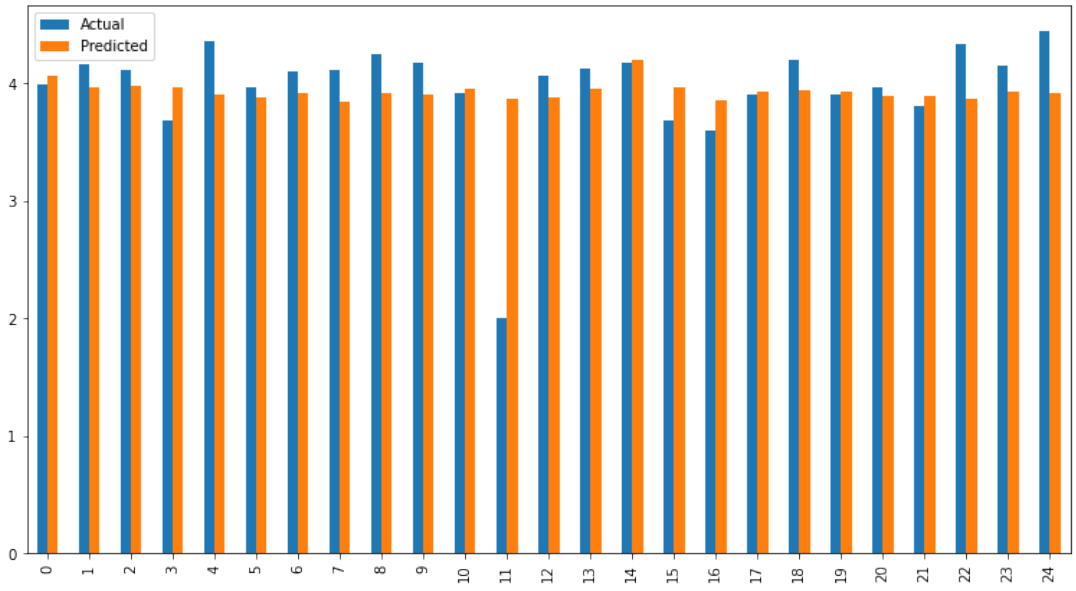
[66]:
```
pred.head(10)
```

[66]:

	Actual	Predicted
0	3.99	4.064338
1	4.16	3.961904
2	4.12	3.975257
3	3.68	3.964341
4	4.36	3.903166
5	3.97	3.878353
6	4.10	3.920054
7	4.12	3.845629
8	4.25	3.920088
9	4.17	3.905307

[67]:
```
# visualise the above comparison result
pred.plot(kind='bar', figsize=(13, 7))
```

[67]: <AxesSubplot:>



[68]:
```
print('MAE:', metrics.mean_absolute_error(y_test, predictions))
print('MSE:', metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
```

```
MAE: 0.2275157380024754
MSE: 0.11575926260147222
RMSE: 0.34023412909564527
```

[ ]:

Conclusion

Model-1

```
MAE: 0.22785560782959874
MSE: 0.11598507711983519
RMSE: 0.34056581907149047
```

Model-2

```
MAE: 0.22401505196756463
MSE: 0.10527048686775198
RMSE: 0.32445413677090323
```

model-3

```
MAE: 0.2275157380024754
MSE: 0.11575926260147222
RMSE: 0.34023412909564527
```

According to the results we can see that, of the three models, comparing, MAE, MSE and RMSE, we found that the best way to treat the data was eliminating the outlier data, having an improvement with models 2 and 3 with respect to the model 1.