CS 418: Introduction to Data Science
Final Project : Rating Analysis and Prediction for Books
(Data Science Pipeline on Books DataSet)
Fall 2019

Project Report

Problem Selection :

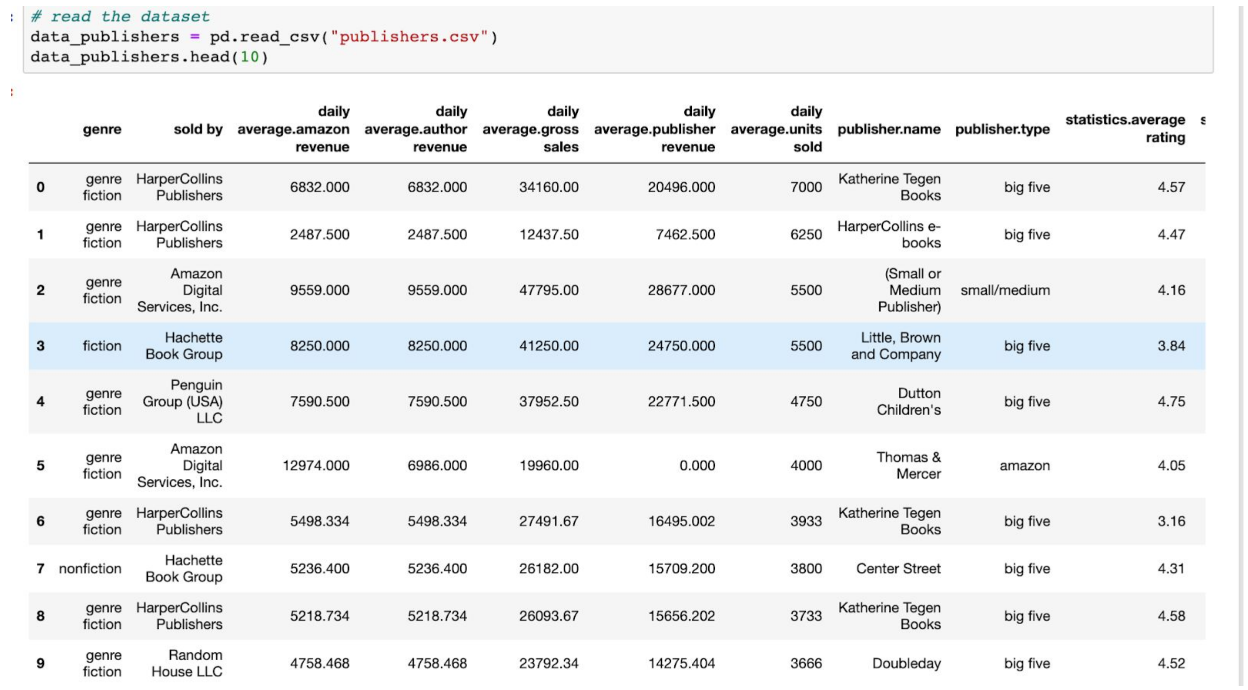
The goal of this project is to analyse the important factors that make a book more popular and most read compared to others and to use this analysis to predict the average rating of a book and to categorize the books into corresponding genres.

Proposed Solution :

Data preprocessing is performed on the raw data to clean the observations and using Descriptive Statistics the importance of each variable is determined.
These conclusions on the variables are used to build a regression model for the prediction of average rating of the books.
Using classification or clustering techniques, the books are categorized into specific genres.

Data Collection :

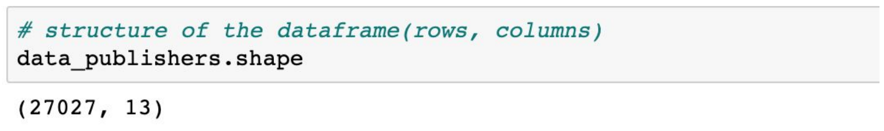
We are using Publishers dataset from CORGIS datasets. The dataset consists of information about different titles spanned across different genres and publishers. It also includes the average rating and overall sales ranking of amazon kindle store.
The variables in the dataset include:
genre, sold_by, daily_average_amazon_revenue, daily_average_author_revenue, daily_average_gross_sales, daily_average_publisher_revenue, daily_average_units_sold, publisher_name, publisher_type, average_rating, sale_price, sales_rank, total_reviews



```
# read the dataset
data_publishers = pd.read_csv("publishers.csv")
data_publishers.head(10)
```

	genre	sold by	daily average.amazon revenue	daily average.author revenue	daily average.gross sales	daily average.publisher revenue	daily average.units sold	publisher.name	publisher.type	statistics.average rating	s
0	genre fiction	HarperCollins Publishers	6832.000	6832.000	34160.00	20496.000	7000	Katherine Tegen Books	big five	4.57	
1	genre fiction	HarperCollins Publishers	2487.500	2487.500	12437.50	7462.500	6250	HarperCollins e-books	big five	4.47	
2	genre fiction	Amazon Digital Services, Inc.	9559.000	9559.000	47795.00	28677.000	5500	(Small or Medium Publisher)	small/medium	4.16	
3	fiction	Hachette Book Group	8250.000	8250.000	41250.00	24750.000	5500	Little, Brown and Company	big five	3.84	
4	genre fiction	Penguin Group (USA) LLC	7590.500	7590.500	37952.50	22771.500	4750	Dutton Children's	big five	4.75	
5	genre fiction	Amazon Digital Services, Inc.	12974.000	6986.000	19960.00	0.000	4000	Thomas & Mercer	amazon	4.05	
6	genre fiction	HarperCollins Publishers	5498.334	5498.334	27491.67	16495.002	3933	Katherine Tegen Books	big five	3.16	
7	nonfiction	Hachette Book Group	5236.400	5236.400	26182.00	15709.200	3800	Center Street	big five	4.31	
8	genre fiction	HarperCollins Publishers	5218.734	5218.734	26093.67	15656.202	3733	Katherine Tegen Books	big five	4.58	
9	genre fiction	Random House LLC	4758.468	4758.468	23792.34	14275.404	3666	Doubleday	big five	4.52	

We have downloaded the publishers.csv file from the CORGIS website and used pandas to read the data from the csv file. The dataset has 27027 observations and 13 features.



```
# structure of the dataframe(rows, columns)
data_publishers.shape
```

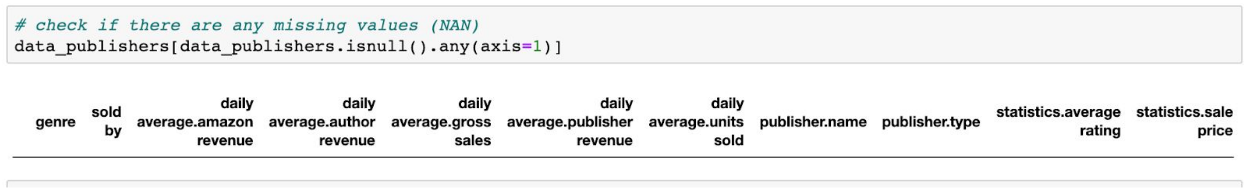
```
(27027, 13)
```

Data Preparation :

As part of Data preparation, we have checked the dataset for missing values, zero values, and the proportion of data corresponding to each genre and removed the data belonging to genres with low proportion.

Missing Values:

We have checked the dataset for missing values (NAN).
There are no missing values in the data set.

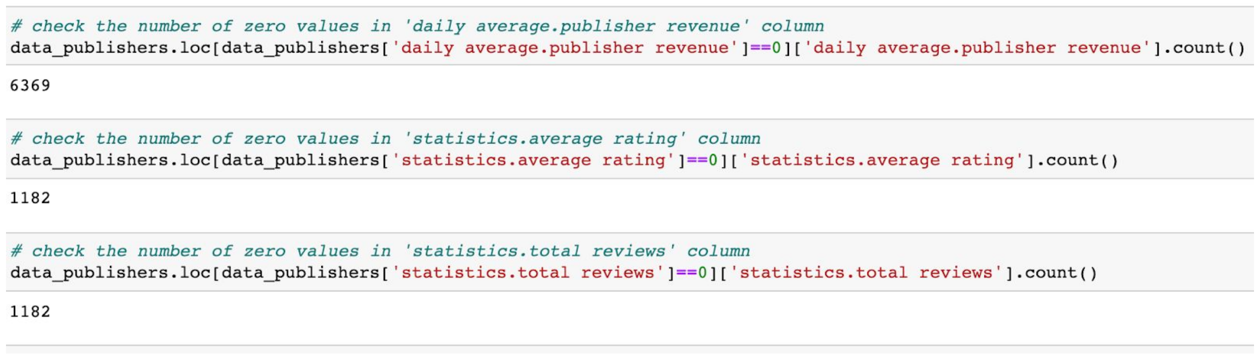


```
# check if there are any missing values (NAN)
data_publishers[data_publishers.isnull().any(axis=1)]
```

genre	sold by	daily average.amazon revenue	daily average.author revenue	daily average.gross sales	daily average.publisher revenue	daily average.units sold	publisher.name	publisher.type	statistics.average rating	statistics.sale price

Zero Values:

We have checked the dataset for zero values.
There are zero values for daily_average_publisher_revenue, statistics.average rating and statistics.total reviews.
Since zero is a valid value for all the three features, we have retained them.



```
# check the number of zero values in 'daily average.publisher revenue' column
data_publishers.loc[data_publishers['daily average.publisher revenue']==0]['daily average.publisher revenue'].count()
```

```
6369
```

```
# check the number of zero values in 'statistics.average rating' column
data_publishers.loc[data_publishers['statistics.average rating']==0]['statistics.average rating'].count()
```

```
1182
```

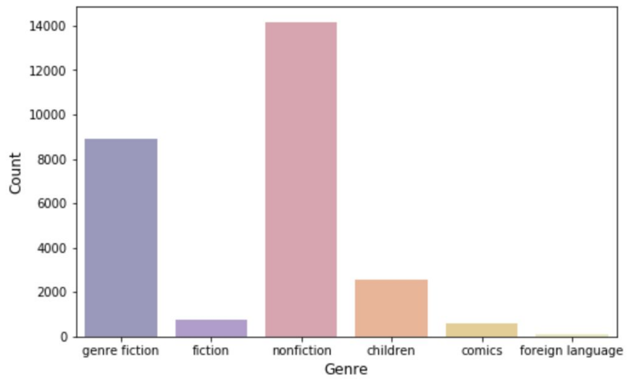
```
# check the number of zero values in 'statistics.total reviews' column
data_publishers.loc[data_publishers['statistics.total reviews']==0]['statistics.total reviews'].count()
```

```
1182
```

<u>Group By genre</u>:

We have analysed the proportion of each genre in the dataset by grouping the dataset by genre.

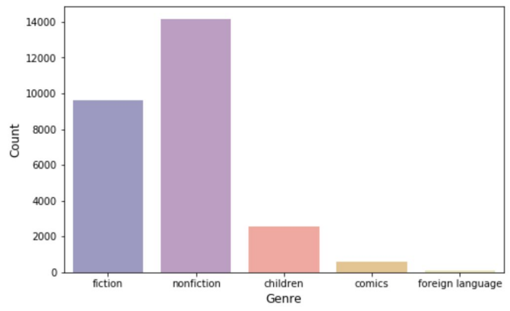
```
genre
children              9.401709
comics                2.101602
fiction               2.712103
foreign language      0.447700
genre fiction        32.941133
nonfiction           52.395752
```



We have replaced ‘genre fiction’ with ‘fiction’ since both are the same.

```
# replace 'genre fiction' with 'fiction' to consider both as same genre
data_publishers_tidy = data_publishers.replace(to_replace = "genre fiction",  value = "fiction")
data_publishers_tidy.head()
```

```
genre
children              9.401709
comics                2.101602
fiction              35.653236
foreign language      0.447700
nonfiction           52.395752
```

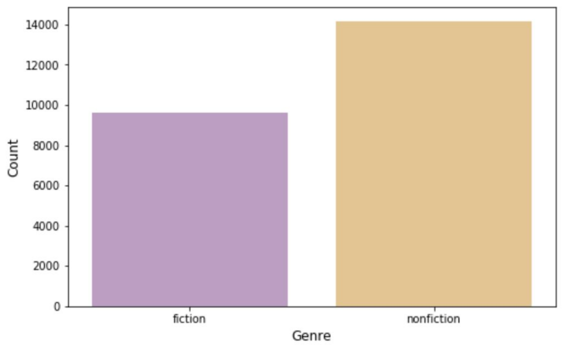


Since 90 percent of the data belong to ‘fiction’ or ‘nonfiction’ and other 10 percent belong to ‘children’, ‘comics’ and ‘foriegn language’, we have dropped the observations belonging to ‘children’, ‘comics’ and ‘foriegn language’ genres.

```
# remove the genres that has very less proportion in the dataframe to avoid class imbalance issues
data_publishers_tidy = data_publishers_tidy.drop(data_publishers_tidy[(data_publishers_tidy['genre'] == 'children') |
                                                                      (data_publishers_tidy['genre'] == 'comics') |
                                                                      (data_publishers_tidy['genre'] == 'foreign languag
                                                                      .index)
data_publishers_tidy.head()
```

The final dataset has only two genres - ‘fiction’ and ‘nonfiction’.

```
genre
fiction        40.492499
nonfiction     59.507501
```



Get Dummies:

We have dropped the two categorical attributes ‘sold_by’ and ‘publisher_name’ since they are the names or ids and do not impact the data analysis
Since we have two categorical attributes, genre and publisher_type we used dummies to convert them to numerical attributes.

```
# dropping sold by and publisher.name since they do not impact the rating
# get the dummy values for genre and publisher.type
data_publishers_tidy = pd.get_dummies(data_publishers_tidy.iloc[:,[0,2,3,4,5,6,8,9,10,11,12]], drop_first = True)
data_publishers_tidy.head()
```

average rating	statistics.sale price	statistics.sales rank	statistics.total reviews	genre_nonfiction	publisher.type_big five	publisher.type_indie	publisher.type_single author	publisher.type_small/medium
4.57	4.88	1	9604	0	1	0	0	0
4.47	1.99	2	450	0	1	0	0	0
4.16	8.69	3	30	0	0	0	0	1
3.84	7.50	3	3747	0	1	0	0	0
4.75	7.99	4	9174	0	1	0	0	0

Data Exploration

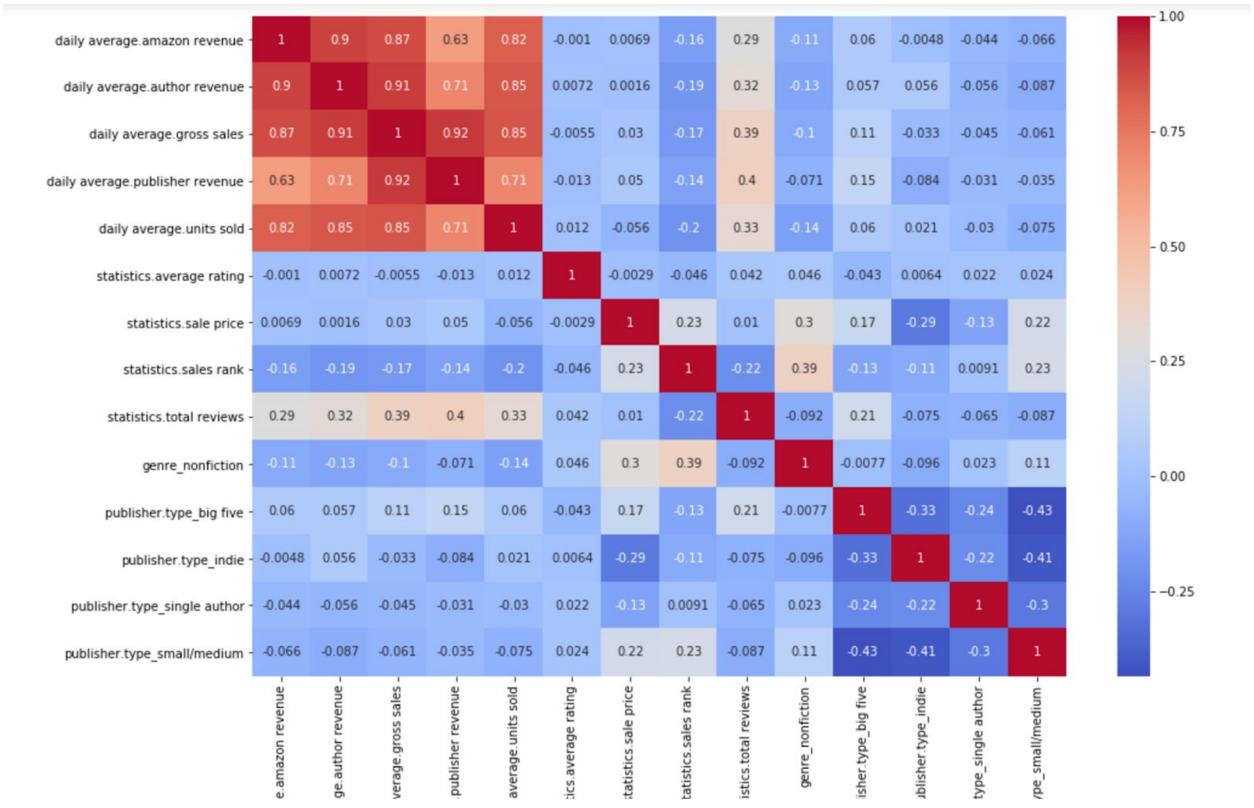
The dataset is a multi dimensional data. To visualize the relationship between the attributes, we need scatter plot matrix and correlation matrix.

Correlation Matrix:

We have used df.corr() to build the correlation matrix. The correlation matrix displays the pair wise correlation coefficients between the variables.
We have extracted the variables with high correlation coefficients with ‘statistics.average rating’ since we want to build a model to predict the average rating.

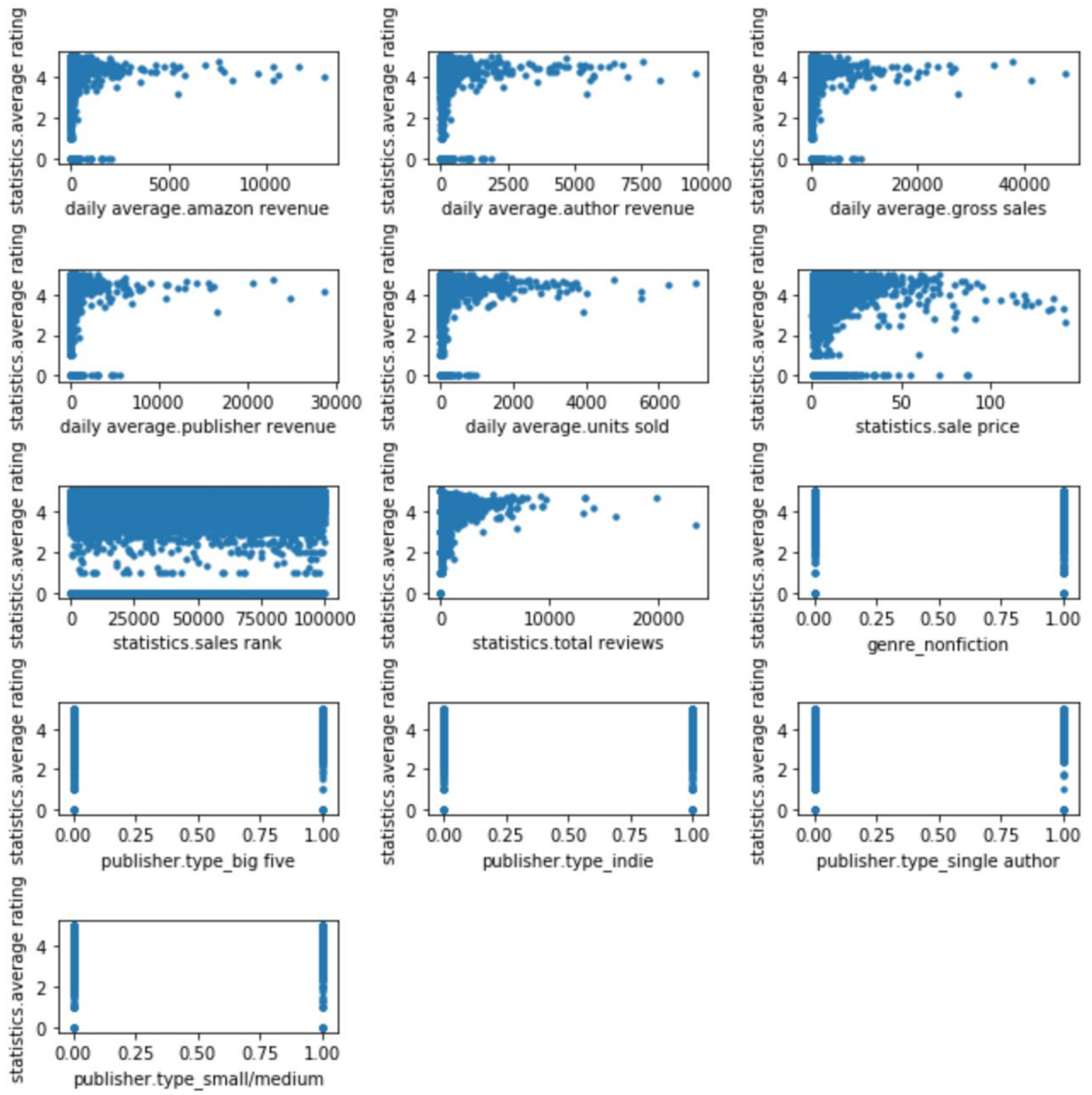
```python
# get the correlation matrix and plot heatmap to check the correlation of statistics.average rating with other variabl
correlation_matrix = data_publishers_tidy.corr()
correlation_features = correlation_matrix.index
plt.figure(figsize=(15,15))
graph = sns.heatmap(data_publishers_tidy[correlation_features].corr(),annot=True,cmap="coolwarm")

# get the variables with correlation > 0.01 as important variables
correlation_y = abs(correlation_matrix["statistics.average rating"])
important_features = correlation_y[correlation_y > 0.01]
print("The important variables to predict average rating of the book are:")
print(important_features)
```

```
The important variables to predict average rating of the book are:
daily average.publisher revenue    0.012713
daily average.units sold           0.011972
statistics.average rating          1.000000
statistics.sales rank              0.045586
statistics.total reviews           0.041710
genre_nonfiction                   0.045907
publisher.type_big five            0.042833
publisher.type_single author       0.022082
publisher.type_small/medium        0.024398
Name: statistics.average rating, dtype: float64
```



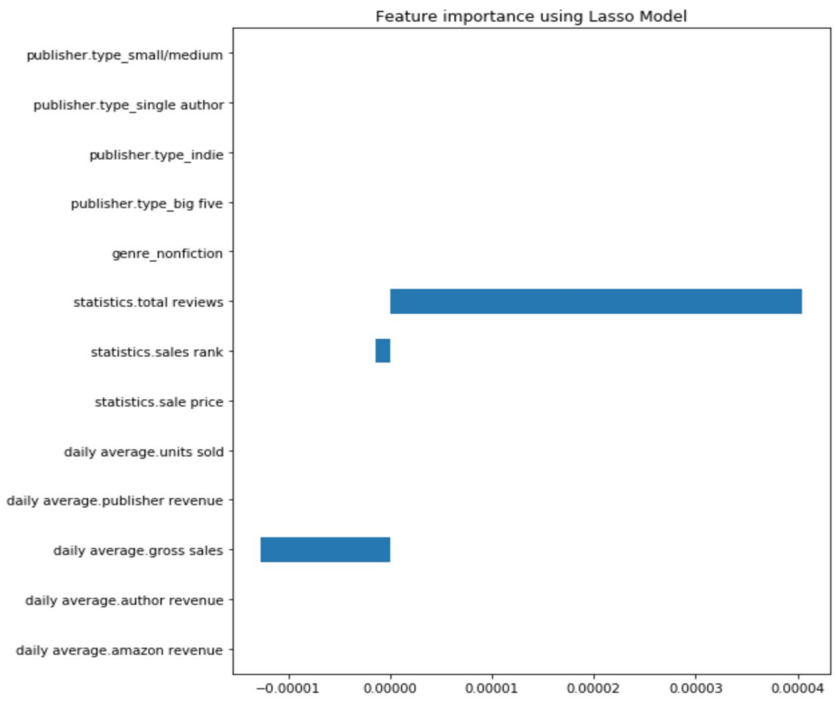
<u>Scatter plots</u>:

We have used df.plot.scatter() to display the scatter plots. The scatter plots displays the pairwise relationship between the variables and ‘statistics.average rating’.



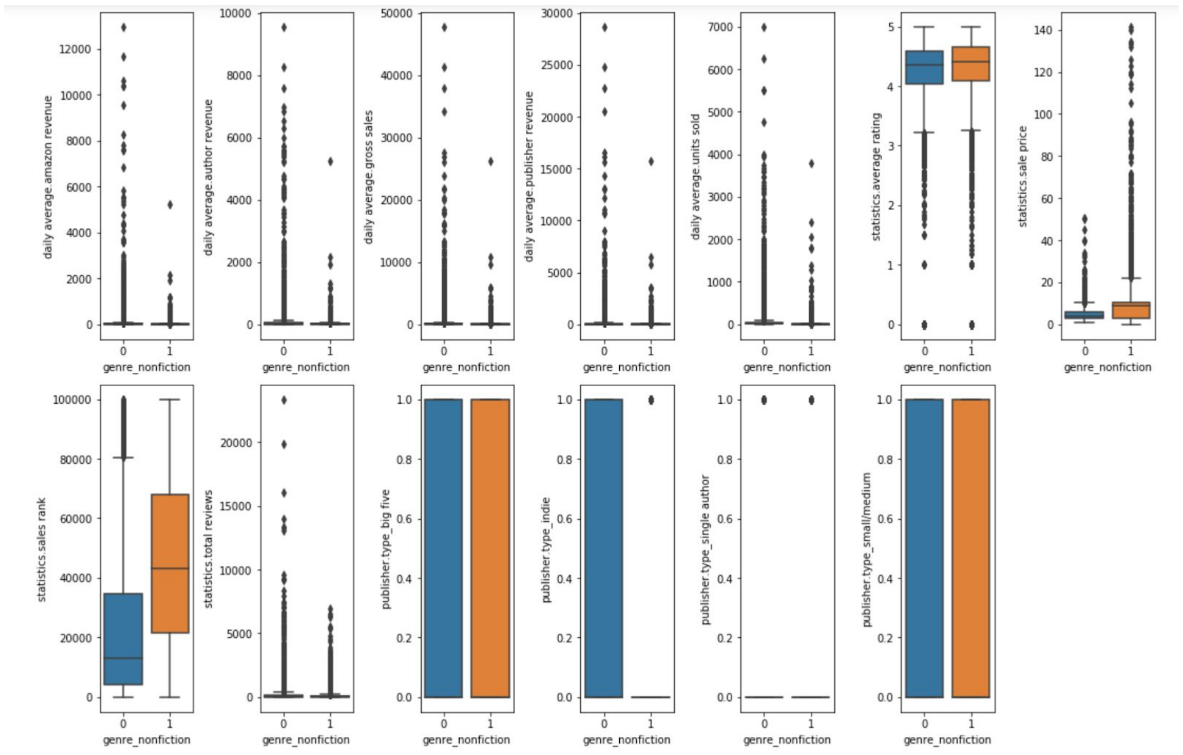
LassoCV:

We have used LassoCV for feature selection for building regression model to predict the 'statistics.average rating'.



<u>Box plots</u>:

We have used sns.boxplot() to build the box plots. The box plots displays the pair wise distribution of data between genre and the other variables. These box plots are used to determine the important variables for clustering of the dataset into genres.



Data Modelling :

The data is partitioned into training, validation and test set. The training set is used to train the models and validation set is used to evaluate the models. The best model on the validation set is used to predict the target values on the test set.

Regression:

The features for the regression model are selected based on the findings from the correlation matrix and the feature importance using LassoCV. The variables 'daily average.publisher revenue', 'daily average.units sold', 'statistics.average rating', 'statistics.sales rank', 'statistics.total reviews', 'genre_nonfiction', 'publisher.type_big five', 'publisher.type_single author', 'publisher.type_small/medium' are the selected features for regression.
Different combinations of these variables are used to build different regression models. The best results were obtained using 'statistics.sales rank', 'statistics.total reviews', 'genre_nonfiction', 'publisher.type_big five', 'publisher.type_single author' and 'publisher.type_small/medium'

Model	**R Square**	**Adjusted R Square**	**Root Mean Square Error**
Linear Regression Model	0.012	0.010	0.939
Ridge Regression Model	0.012	0.010	0.939
Decision Tree Regressor Model	0.536	0.536	0.639
Gradient Boosting Regressor Model	0.672	0.672	0.537
Random Forest Regressor Model	0.711	0.711	0.504
Random Forest Regressor Model on Test Set	0.721	0.721	0.495

Five regression models were performed: Linear, Ridge, Decision Tree Regressor, Gradient Boosting Regressor and Random Forest Regressor.
The R-square, Adjusted R-square and Root Mean Square Error values for all the models are listed in the table above.

Out of the five models, Random Forest Regressor performed better in predicting the average rating values both in terms of Adjusted R Square and Root Mean Square Error.

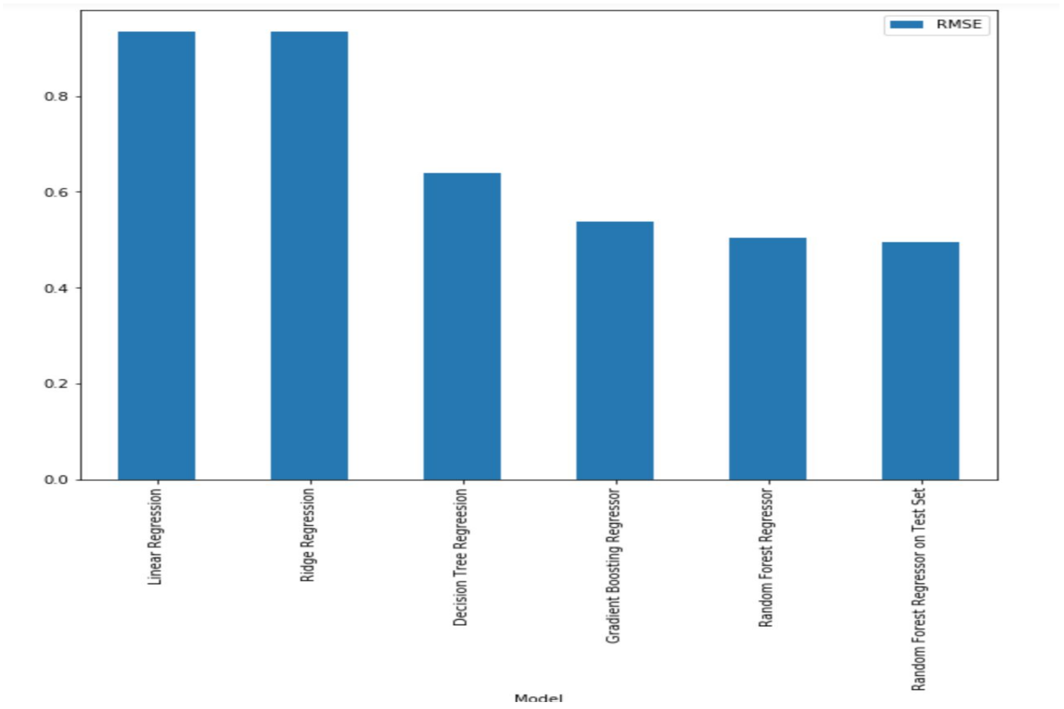
The r squared value measures what proportion of the variance in the response variable is explained by the model. A higher r squared correlates to a better model. When there are multiple predictor variables used, however, the adjusted R square value must be used. This is done so that the value of the predictor variables can be taken into account. If a predictor variable does not improve the model, it will result in a more complex model which can lead to overfitting. The adjusted R square penalizes any variables that are unnecessarily added and do not improve the model.

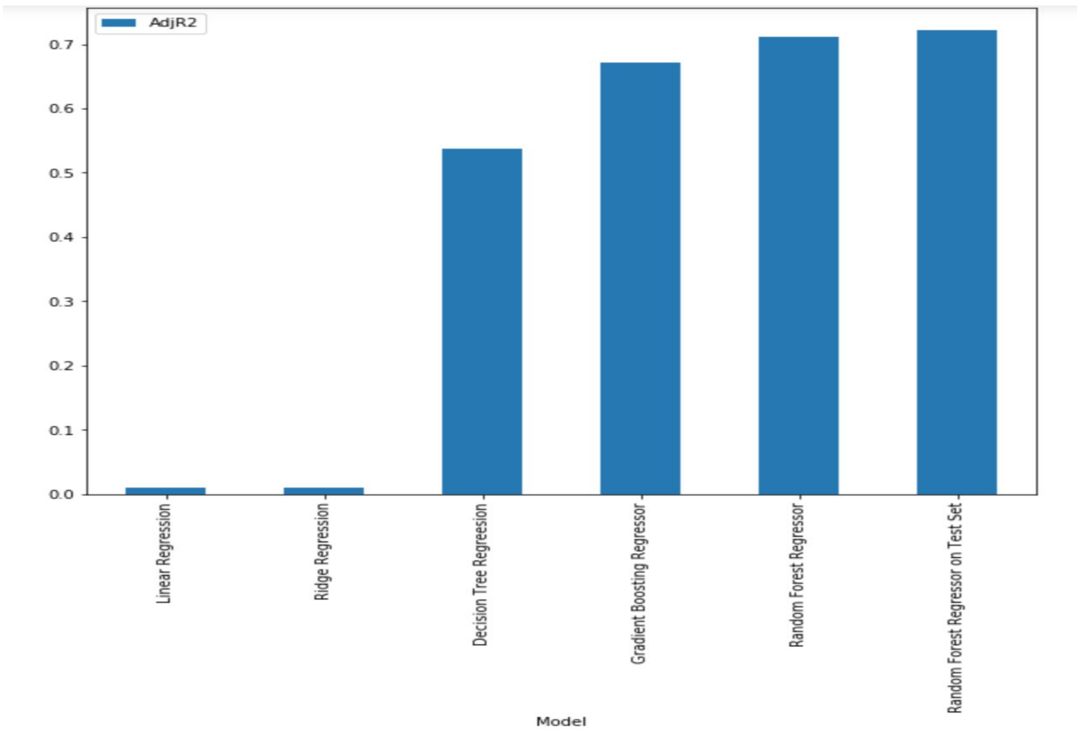
Predictions on Test Set:

The best model Random Forest Regressor is used to predict the average rating values on the test set.

Results:

The Adjusted R Square on the test set is 0.721 and the Root Mean Square Error is 0.495. The below graphs show the compared Adjusted R square and Root Mean Square values of different models on the validation set and the best model on Test set.





Classification:

A few classification models, all a variant of Decision Trees, were trained using the data with the goal of predicting the genre of a book. These models include:

- Basic Decision Tree using the entire feature set
- Basic Decision Tree using a subset of features
- Bagging Ensemble Method with Decision Tree as a base estimator
- Adaboost Ensemble Method with Decision Tree as a base estimator
- Random Forest Ensemble Method with Decision Tree as a base estimator

All these models employed entropy as their node splitting criterion.

Model	**Accuracy**	**Precision**	**F1 Score**
Basic Decision Tree - All Features	0.73870	0.72847225	0.73012765
Basic Decision Tree - Filtered Features	0.734243	0.723778225	0.725329835
Bagging - All Features	0.782825	0.774126725	0.77649256
Adaboost - All Features	0.737920	0.727683025	0.729340125
Random Forest - All Features	0.775472	0.767162875	0.769499605

Out of the five models, Bagging with a base Decision Tree Classifier performed better in predicting the genre with better average statistics across the board

The consistent accuracy, precision and F1 score across all models points to no signs of class imbalance which is to be expected given the almost 55-45 split of genres in the dataset. All models remained within the 70% bracket for performance statistics with bagging just barely coming out on top in all areas.

Predictions on Test Set:

The best model Bagging Ensemble method with a base Decision Tree Classifier is used to predict the genre on the test set.

Results:

On the test set, our best model achieved the following statistics.

```
Accuracy -     0.7783613445378151
Error -        0.22163865546218486
Precision -    [0.69985775, 0.84081479] => Average - 0.77033627
Recall -       [0.7776607, 0.778826 ]   => Average - 0.77824335
F1 Score -     [0.73671076, 0.80863414] => Average - 0.77267245
```

We can conclude that model performs fairly well given the feature set however a higher degree of accuracy and precision may be desired given the use case of the model.

Clustering:

Various clustering models were implemented and run on the entire data set. These models include:

- Hierarchical Single Linkage with Euclidean, Manhattan, Minkowski, or Cosine distances.
- Hierarchical Complete Linkage with Euclidean, Manhattan, Minkowski, or Cosine distances.
- Hierarchical Average Linkage with Euclidean, Manhattan, Minkowski, or Cosine distances.
- K-Means Clustering with Random and k-means++ initial clusters
- DBSCAN with different epsilon distances and min points

Different combinations the feature set was used on the above models. The best models were hierarchical complete with cosine distance, hierarchical average with cosine distance and K-Means. The version of these models featured the included code utilise the entire feature set or a subset of the features limited to 'daily average.gross sales', 'daily average.publisher revenue', 'daily average.units sold', 'statistics.average rating', 'statistics.sale price', 'statistics.total reviews', 'publisher.type_single author', 'publisher.type_small/medium'

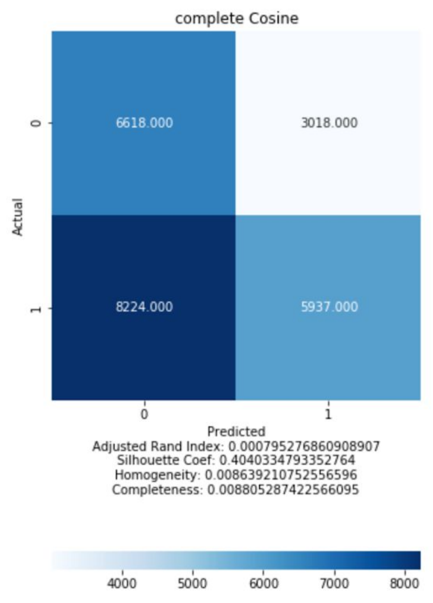
Model	**Silhouette Coefficient**	**Adjusted Rand Index**
Hierarchical Complete Cosine - Filtered Features	0.396	0.015
Hierarchical Average Cosine - Filtered Features	0.564	-0.0016
K-Means Random - Filtered Features	.3384	-0.00109
K-Means k-Means++ - Filtered Features	0.0918	0.00158

For most models, the limited feature set mentioned above resulted in a more desirable modeling of clusters.
Out of these models, Hierarchical Complete with cosine linkage performed better in terms of matching the true cluster silhouette coefficient while having comparable or better adjusted rand index to its peers.

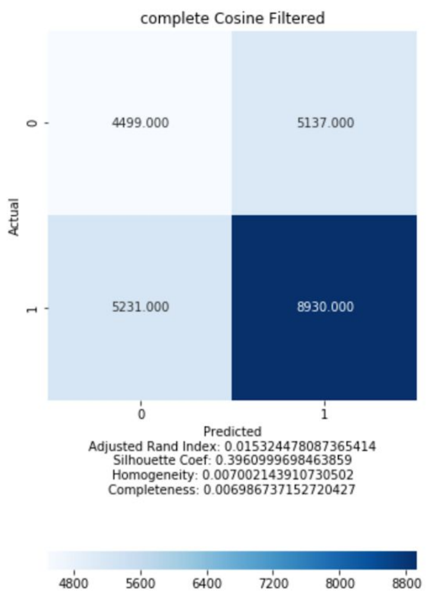
The other models not featured in the above table suffered from assigning too many observations to the same class i.e creating a single large cluster with a very high silhouette coefficient . DBSCAN resulted in 8-10 clusters for each of its variations.

Results:

Clustering analysis on this data shows that the feature set provided may have no concrete dissimilarity of relationships amongst themselves strong enough to put them into meaningful clusters. Further analysis of the cluster labels mapped back to observations showed no significant separation of the values in the features that can differentiate them as fiction or nonfiction. The features or atleast the combinations present here may not be the strongest indicators of what genre a book falls into i.e; books of these genres share many of these above traits to a degree where the above models can only form trivial associations or clusters. The silhouette coefficient of the true clusters seems to agree with a loose association among observations if any.



Adjusted Rand Index: 0.000795276860908907
Silhouette Coef: 0.4040334793352764
Homogeneity: 0.008639210752556596
Completeness: 0.008805287422566095



Adjusted Rand Index: 0.015324478087365414
Silhouette Coef: 0.3960999698463859
Homogeneity: 0.007002143910730502
Completeness: 0.006986737152720427

True Clusters

[53]:
```
silhouette_coefficient =  silhouette_score(clusterFeatures, clusterClass, metric='euclidean')
print(silhouette_coefficient)
```

```
0.1224181820045522
```