CS839 Project Stage 4 Report

Sreejita Dutta, Deepanshu Gera, Rahul Jayan

I. Dataset and Data Merging

In the last project stage, 802 tuple pairs had survived the blocking stage. As we hadn't applied the matcher M on *all* the tuple pairs in the previous stage, we had to do that this time to obtain the match table. The match table is stored in *predictions_all_with_features.csv* file. In the match table, "ltable" represents Amazon data, "rtable" represents Walmart data and "*predict_labels"* represent the actual prediction (whether it is a match of not).

For combining the data, we did not use any other dataset except the tables that we had already, that is, Amazon books data and Walmart books data.

Steps for Data Merging:

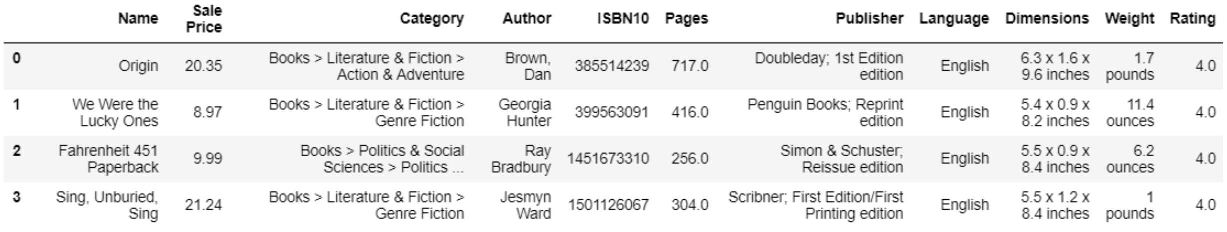
1) A match table was obtained after applying the matcher M to candidate tuple pairs.
2) Using the match table, we first obtain the predicted matches from amazon_id -> walmart_id and vice versa. This is done as one book in amazon dataset can match with multiple books from walmart dataset.
3) As there can be some transitive matches, the next step was to create a cluster of all the matches.
4) The cluster contains 2 lists (list of amazon books and walmart books). Each entity in a cluster is a match with all the other entities.
5) Once we have all the clusters of matches, we apply the following rules for merging data:
   a) For the *Name, Author* of the book, select the one with the longest string. As there cannot be any null value in Name/Author, we do not have to manage NaN's.
   b) For the *Sale price* and *Pages*, we selected the highest value. In certain cases, some book prices were subjected to seasonal discounts. Our rationale behind choosing the max price was that it was more indicative of the price of the book consistent throughout the year.
   c) For *Category, Publisher, Language, Dimesions and ISBN,* we felt that Amazon data was cleaner. Hence, for these attributes, we preferred Amazon dataset. Although, if these values weren't available in Amazon's data, we use the Walmart dataset as our source of input.
   d) For *Ratings,* we use the minimum of all the values.
   e) Attribute *Weight* was only present in Amazon's dataset. Hence it is directly carried over.
   f) In case the attribute value is missing from all the entities of the cluster, we input NaN.

II. Statistics of Merged Table

The final dataset contains 485 tuples.

The schema of this table includes the following attributes:
'Name', 'Sale Price', 'Category', 'Author', 'ISBN10', 'Pages', 'Publisher', 'Language', 'Dimensions', 'Weight', 'Rating'

Here are four representative tuples in the table.



	Name	Sale Price	Category	Author	ISBN10	Pages	Publisher	Language	Dimensions	Weight	Rating
0	Origin	20.35	Books > Literature & Fiction > Action & Adventure	Brown, Dan	385514239	717.0	Doubleday; 1st Edition edition	English	6.3 x 1.6 x 9.6 inches	1.7 pounds	4.0
1	We Were the Lucky Ones	8.97	Books > Literature & Fiction > Genre Fiction	Georgia Hunter	399563091	416.0	Penguin Books; Reprint edition	English	5.4 x 0.9 x 8.2 inches	11.4 ounces	4.0
2	Fahrenheit 451 Paperback	9.99	Books > Politics & Social Sciences > Politics ...	Ray Bradbury	1451673310	256.0	Simon & Schuster; Reissue edition	English	5.5 x 0.9 x 8.4 inches	6.2 ounces	4.0
3	Sing, Unburied, Sing	21.24	Books > Literature & Fiction > Genre Fiction	Jesmyn Ward	1501126067	304.0	Scribner; First Edition/First Printing edition	English	5.5 x 1.2 x 8.4 inches	1 pounds	4.0

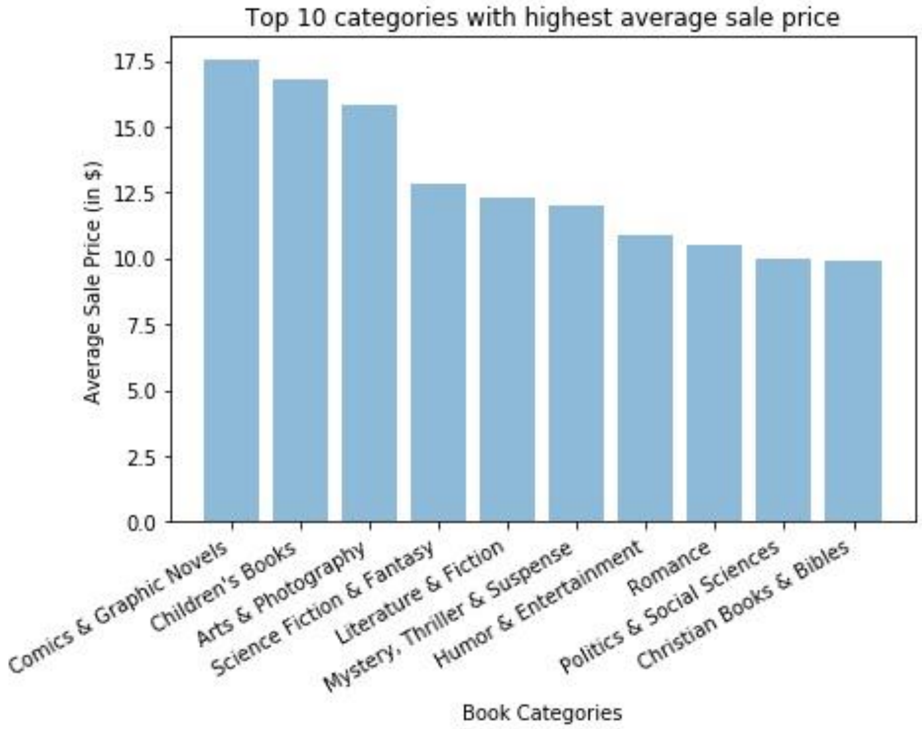
III. Data analysis: tasks, steps and conclusions

We perform **OLAP-style** exploration for analyzing our merged dataset. This involves the four most common operations used in OLAP: roll-up, drill-down, slice and dice.

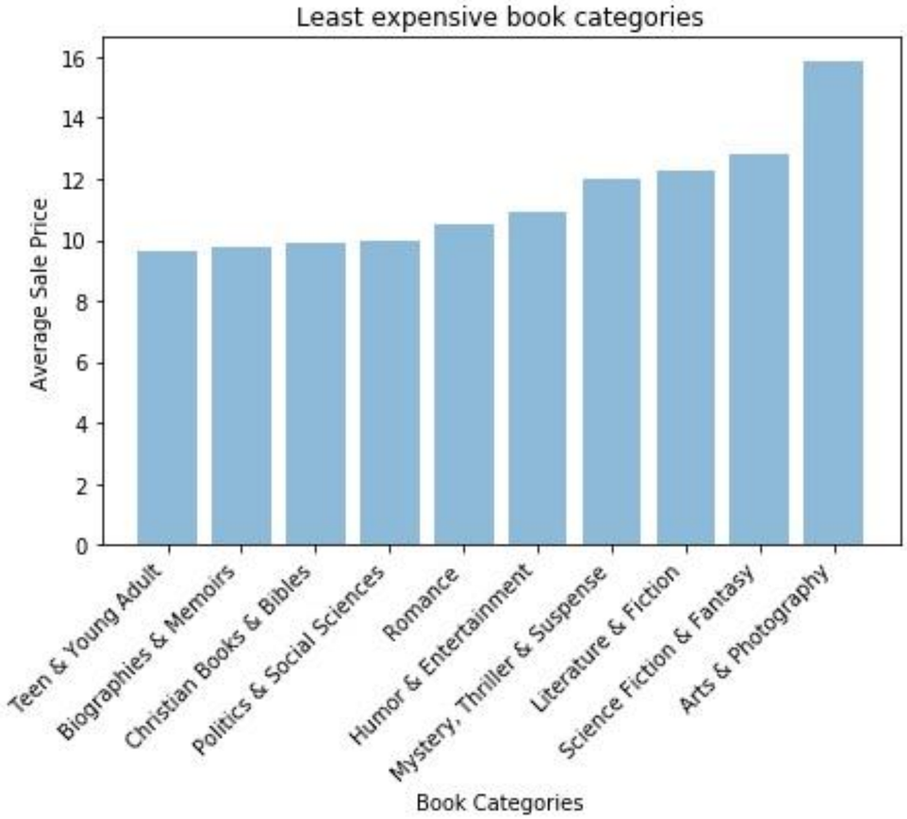
Based on our dataset, we aim to answer the following questions:

(1) What is the average price of books in each category?

We group the sale price of the books in each of the 26 categories of books. Since representing all these groups in one graph is not as visually informative, we split the graph into two. One graph shows the 10 categories with highest average price and the next one depicting bottom 10 categories.



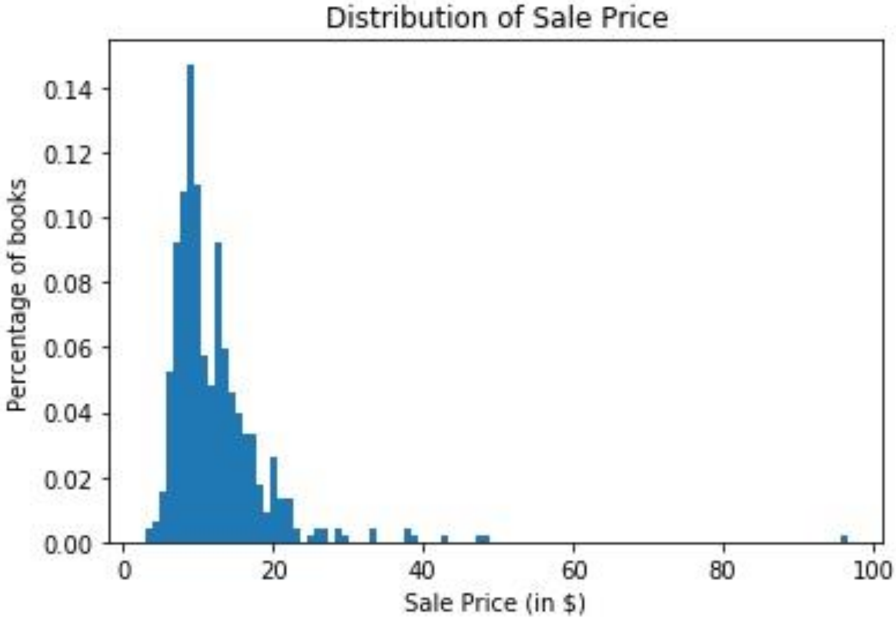
Among the different book categories, we find that there are two categories which has higher average sale price than all the others – Comics & Graphics Novels and Children's books



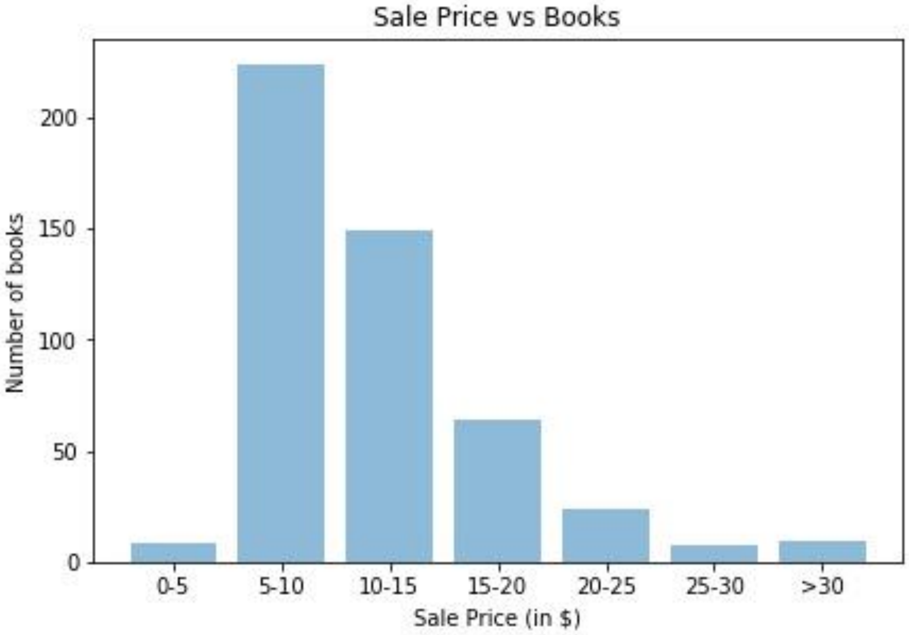
We infer that biographies, Christian literature and books catering to Teens are the least priced ones.

(2) How are books distributed amongst various price ranges?

The average price range of books gives us valuable information on how to price new books in the future. This plot gives us an idea of how the distribution looks when plotted against price attribute.

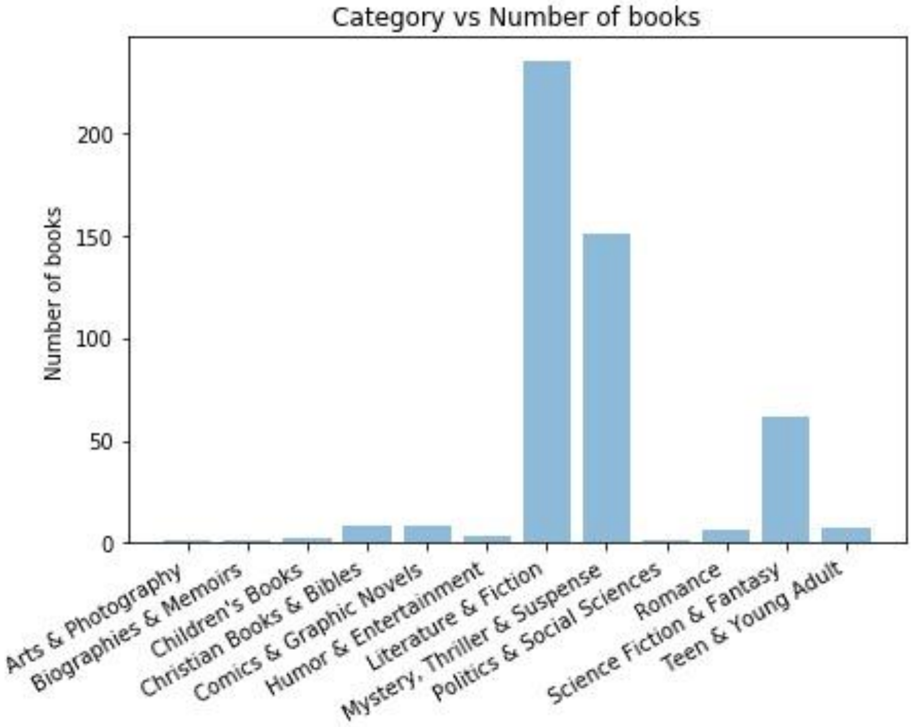


To get a much-detailed price distribution, we grouped price into different bins and then plotted against the number of books. We grouped sale price into 7 categories given below. We grouped all the books with price greater than 30$ into one category as it is an under represented one. From the plot we can infer that most of the books are priced between $ 5 and $10.

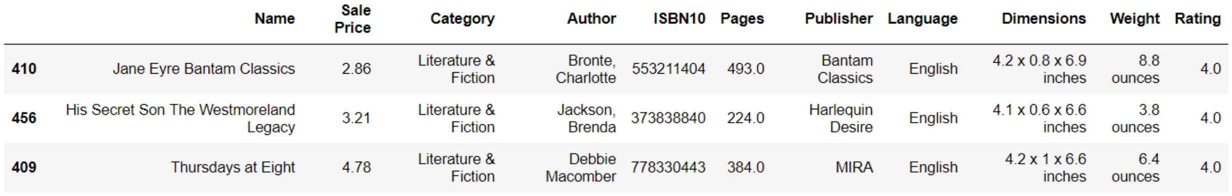


(3) What are the names of reasonably priced books available in the category with highest number of books?

To figure out reasonably priced books in a category, we must infer the general trend. From the below graph we can infer that Literature and Fiction is the most popular category where most number of books are being sold.



From the above information, we group all the entries with category as ‘Literature & Fiction’ and then use this information to find the least priced ones. This kind of inference is useful for both publishers and the casual readers. The publishers and sellers can get an idea of support price which indicates how low can the books be priced and for the casual reader a cheap book in his favorite genre.

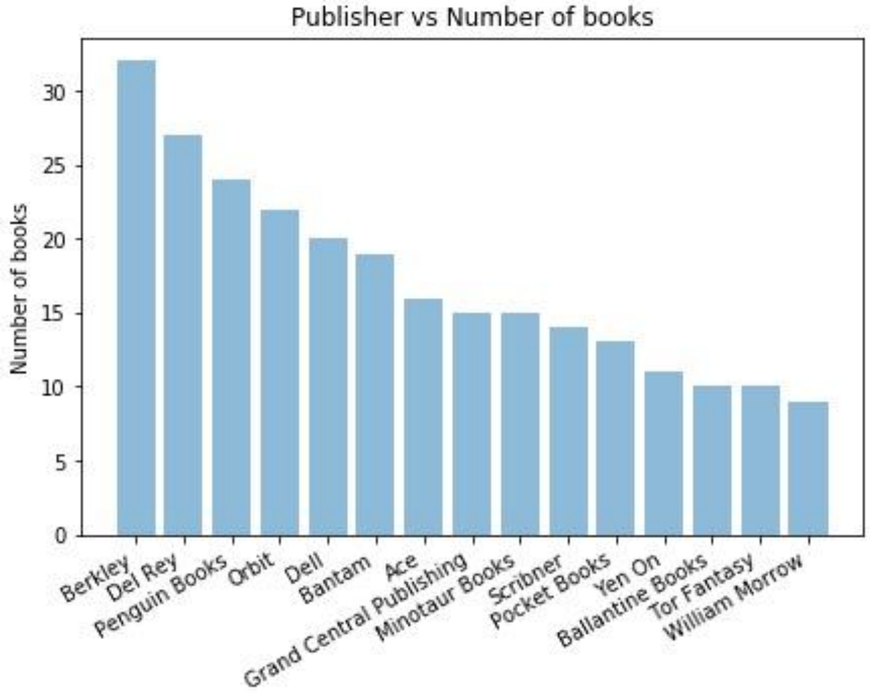


	Name	Sale Price	Category	Author	ISBN10	Pages	Publisher	Language	Dimensions	Weight	Rating
410	Jane Eyre Bantam Classics	2.86	Literature & Fiction	Bronte, Charlotte	553211404	493.0	Bantam Classics	English	4.2 x 0.8 x 6.9 inches	8.8 ounces	4.0
456	His Secret Son The Westmoreland Legacy	3.21	Literature & Fiction	Jackson, Brenda	373838840	224.0	Harlequin Desire	English	4.1 x 0.6 x 6.6 inches	3.8 ounces	4.0
409	Thursdays at Eight	4.78	Literature & Fiction	Debbie Macomber	778330443	384.0	MIRA	English	4.2 x 1 x 6.6 inches	6.4 ounces	4.0

(4) Publisher vs Price

Another major trend which we want to infer is the relationship between publishers and sale price. We want to figure out if there exist some publishers who sells books at a higher price than the rest of the publishers.

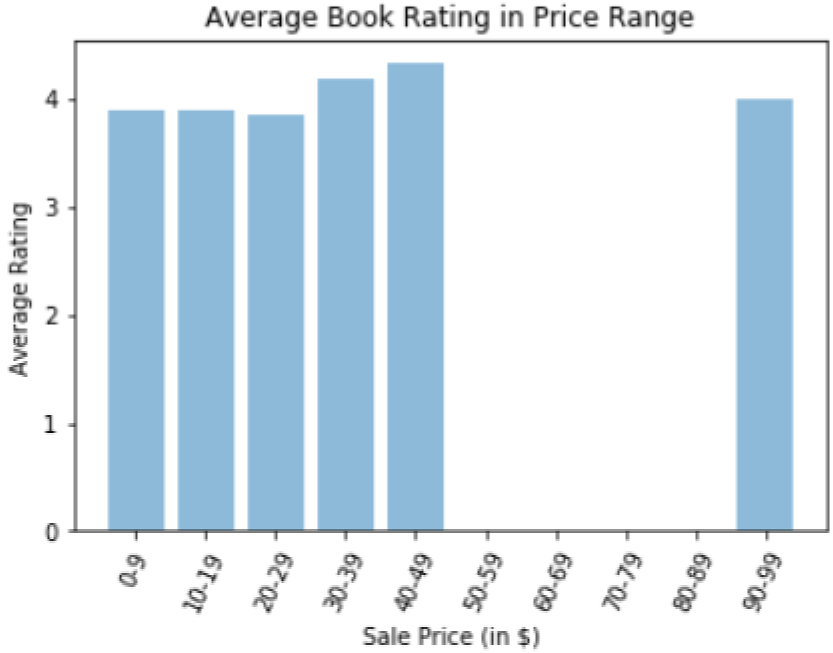
From the plot we can infer that the Berkley Publishing and Del Rey Books are the two publishing houses with higher average sale price than the rest of the publishers.



(5) What is the average rating of books in each price range?

We group the Sale Price of books in 10$ intervals between 0 – 99 $ since all our books fall into this price range. For each price range, we compute the average rating of the books that are sold in that price range.

Here, we can observe that most books are sold at a price below 50$ and the average rating of books is highest in the 40-49$ price range and lowest in the 20-29$ price range.



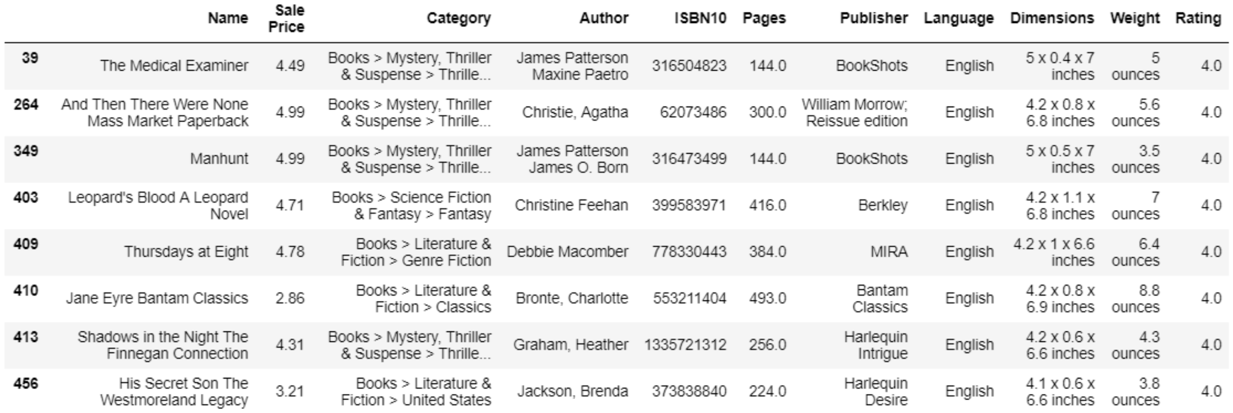
(6) Which authors receive high ratings for their books consistently?

We aim to find a list of favorite authors from the given dataset. For answering this question, we decided to obtain the names of authors who have received a rating of at least 4 stars out of 5 for more than 7 books.

First, we normalize the author names (as they are in no specific format). Then, we find all the authors that satisfy the above conditions. Here is the list in decreasing order of popularity.

Stephen King, Nora Roberts, Lee Child, Louise Penny, Patricia Briggs, Janet Evanovich, Brandon Sanderson, James Patterson

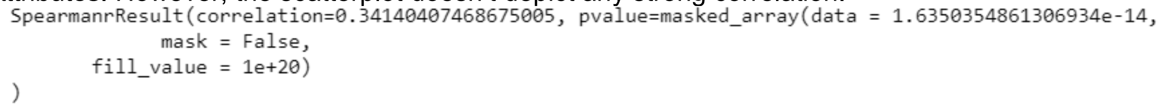
(7) Which books have the potential to offer greatest value for money?
This information could potentially be useful to customers who want to buy a high rating book (at least 4 starts out of 5) at a reasonable price of less than 5$.
From the obtained list of 8 books qualifying this condition, it is interesting to note that there are two books written by *James Patterson* (who is one of the popular authors from 6.) and majority of the books fall under the *Thriller & Suspense* category.



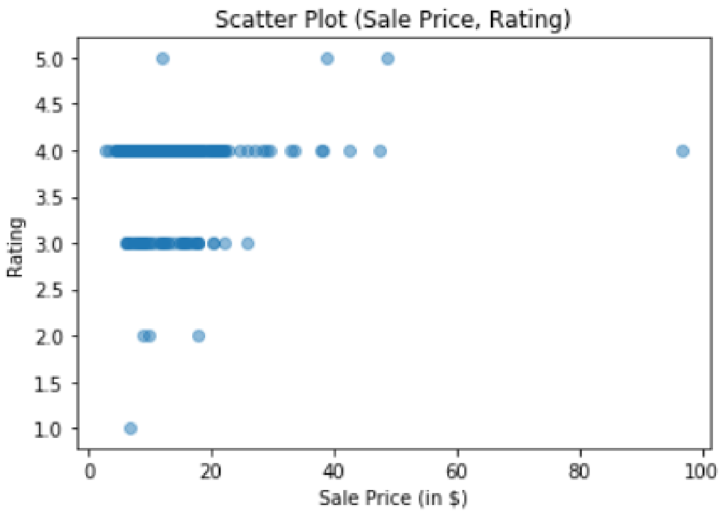
	Name	Sale Price	Category	Author	ISBN10	Pages	Publisher	Language	Dimensions	Weight	Rating
39	The Medical Examiner	4.49	Books > Mystery, Thriller & Suspense > Thrille...	James Patterson Maxine Paetro	316504823	144.0	BookShots	English	5 x 0.4 x 7 inches	5 ounces	4.0
264	And Then There Were None Mass Market Paperback	4.99	Books > Mystery, Thriller & Suspense > Thrille...	Christie, Agatha	62073486	300.0	William Morrow; Reissue edition	English	4.2 x 0.8 x 6.8 inches	5.6 ounces	4.0
349	Manhunt	4.99	Books > Mystery, Thriller & Suspense > Thrille...	James Patterson James O. Born	316473499	144.0	BookShots	English	5 x 0.5 x 7 inches	3.5 ounces	4.0
403	Leopard's Blood A Leopard Novel	4.71	Books > Science Fiction & Fantasy > Fantasy	Christine Feehan	399583971	416.0	Berkley	English	4.2 x 1.1 x 6.8 inches	7 ounces	4.0
409	Thursdays at Eight	4.78	Books > Literature & Fiction > Genre Fiction	Debbie Macomber	778330443	384.0	MIRA	English	4.2 x 1 x 6.6 inches	6.4 ounces	4.0
410	Jane Eyre Bantam Classics	2.86	Books > Literature & Fiction > Classics	Bronte, Charlotte	553211404	493.0	Bantam Classics	English	4.2 x 0.8 x 6.9 inches	8.8 ounces	4.0
413	Shadows in the Night The Finnegan Connection	4.31	Books > Mystery, Thriller & Suspense > Thrille...	Graham, Heather	1335721312	256.0	Harlequin Intrigue	English	4.2 x 0.6 x 6.6 inches	4.3 ounces	4.0
456	His Secret Son The Westmoreland Legacy	3.21	Books > Literature & Fiction > United States	Jackson, Brenda	373838840	224.0	Harlequin Desire	English	4.1 x 0.6 x 6.6 inches	3.8 ounces	4.0

We also compute Spearman Rank correlation coefficient and plot Scatter plots to find if any **correlation** exists between:

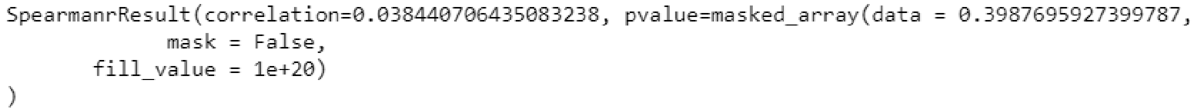
- 'Sale Price' and 'Rating': This is to figure out if highly rated books are generally priced higher. The result obtained shows that there exists some form of correlation between the two attributes. However, the scatterplot doesn't depict any strong correlation.



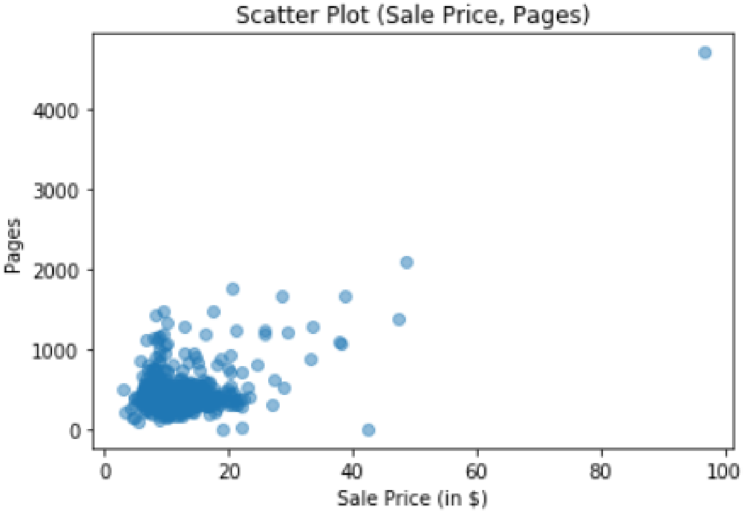
```
SpearmanrResult(correlation=0.34140407468675005, pvalue=masked_array(data = 1.6350354861306934e-14,
             mask = False,
       fill_value = 1e+20)
)
```



- 'Sale Price' and 'Pages': This is to understand if there is any relation between larger books and higher prices. The result shows that there doesn't seem to be any correlation between the two variables.



```
SpearmanrResult(correlation=0.038440706435083238, pvalue=masked_array(data = 0.3987695927399787,
             mask = False,
       fill_value = 1e+20)
)
```



Some problems in our data analysis is listed as follows.

- **Author names are not entered in any specific order or format.**
  The names of authors are not entered in any specific format like <*First Name, Last Name*>. For example, Dan Brown is entered in two ways: i) Dan Brown ii) Brown, Dan. Furthermore, there is no clear way in which multiple authors are separated. Due to these issues, extra steps of analysis and transformation had to be performed before utilizing author data.

- **Preprocessing is required for certain fields to perform any analysis.**
  Attributes like "Category" and "Publisher" have extraneous data. So, they needed to be preprocessed to avoid interference with the analysis tasks. For example, some publisher names were surrounded by texts indicating the print type or the print edition.

IV. Future work

We can perform various other analysis task on this dataset like classification, clustering, association rule mining, anomaly detection, and many other OLAP queries, provided the size of data is sufficient. Examples of some of these are:

(1) **Supervised learning:** In our dataset, we have certain tuples where the Rating of the book is not present. Given sufficient training data, we could train a classifier to predict the rating of a book. This could also be helpful in predicting the ratings of new books.

(2) **Clustering:** We could cluster books based on genre, author, price and rating. This might be useful in certain scenarios. For example, books within the same cluster might appeal to a certain demographic.

V. Code for merging the tables

create_merged_data

May 9, 2018

```
In [203]: import pandas as pd
          import numpy as np
          import csv
```

```
In [204]: M = pd.read_csv("predictions_all.csv")
          left = pd.read_csv("ltable_amazon.csv",encoding = "ISO-8859-1")
          right = pd.read_csv("rtable_walmart.csv",encoding = "ISO-8859-1")
```

```
In [206]: # amazon->wamart matches
          # wamart->amazon matches
          amazon_to_walmart_mapper = {}
          walmart_to_amazon_mapper = {}

          for index, row in left.iterrows():
              left_id = row['id']
              #Find list of matches
              match = M.loc[(M['ltable_id'] == left_id)  & (M['predicted_labels'] == 1)]
              if match.empty:
                  #No match exists for walmart ids add key-> emptylist
                  amazon_to_walmart_mapper[left_id] = []
              else:
                  # Match exist
                  match_list= match.iloc[:, 2].tolist()
                  amazon_to_walmart_mapper[left_id] = match_list

          # Do the same thing for walmart dataset
          for index, row in right.iterrows():
              right_id = row['id']
              #Find list of matches
              match = M.loc[(M['rtable_id'] == right_id) & (M['predicted_labels'] == 1)]
              if match.empty:
                  #No match exists for walmart ids add key-> emptylist
                  walmart_to_amazon_mapper[right_id] = []
              else:
                  # Match exist
                  match_list= match.iloc[:, 1].tolist()
                  walmart_to_amazon_mapper[right_id] = match_list
```

```
In [207]: class Cluster:
              wids = []
              aids = []

              def __init__(self, aids, wids):
                  self.aids = aids
                  self.wids = wids
```

```
In [208]: #create list to manage duplicates
          wids_touched = []
          aids_touched = []
          clusters = [] #List of clusters

          for key, values in amazon_to_walmart_mapper.items():
              aids = []
              wids = []
              if key not in aids_touched:
                  aids.append(key)
                  aids_touched.append(key)
                  wids.extend(values)
                  wids_touched.extend(values)
                  for w_id in values:
                      if w_id in walmart_to_amazon_mapper:
                          a_list = walmart_to_amazon_mapper[w_id]
                          for val in a_list:
                              if val not in aids_touched:
                                  aids_touched.append(val)
                                  aids.append(val)
              clusters.append(Cluster(list(set(aids)),list(set(wids))))
```

```
In [211]: # Now that we have the custers of matches, let's start merging
          # Name - MaxLength
          # Price - Exists/Max
          # Category - Amazon Category
          # Author - MaxLength
          # ISBN - Exists/Amazon has preference
          # Pages - Max
          # Publisher - Exists/ Amazon has preference
          # Language - exists/Amazon
          # Dimensions - esists/Amazon has preference
          # Weight - Amazon
          # Rating - MinRating
          #M.loc[(M['rtable_id'] == right_id) & (M['predicted_labels'] == 1)]
          with open("merged_table.csv", "w", newline='') as csv_file:
              writer = csv.writer(csv_file, delimiter=',')
              writer.writerow(["Name","Sale Price","Category","Author","ISBN10","Pages",
                               "Publisher","Language","Dimensions","Weight","Rating"])
              for cluster in clusters:
```

```
max_name = ""
max_price = 0
category = ""
max_author = ""
isbn = ""
max_pages = 0
publisher = ""
language = ""
dimensions = ""
weight = ""
rating = 100
if cluster.aids and cluster.wids:

    for a_id in cluster.aids:
        if len(left['Name'][a_id]) > len(max_name):
            max_name = left['Name'][a_id]
        if np.isnan(left['Sale Price'][a_id]) == False and
        float(left['Sale Price'][a_id]) > float(max_price):
            max_price = float(left['Sale Price'][a_id])
        if len(left['Category'][a_id]) > len(category):
            category = left['Category'][a_id]
        if len(left['Author'][a_id]) > len(max_author):
            max_author = left['Author'][a_id]
        if len(isbn) == 0:
            isbn = left['ISBN10'][a_id]
        if np.isnan(left['Pages'][a_id]) == False and
        int(left['Pages'][a_id]) > int(max_pages):
            max_pages = int(left['Pages'][a_id])
        if len(publisher) == 0:
            publisher = left['Publisher'][a_id]
        if len(language) == 0:
            language = left['Language'][a_id]
        if len(dimensions) == 0 and left['Dimensions'][a_id] != "nan":
            dimensions = left['Dimensions'][a_id]
            if isinstance(dimensions, str) == False: dimensions = ""
        if len(weight) == 0:
            weight = left['Weight'][a_id]
        if np.isnan(left['Rating'][a_id]) == False and
        int(left['Rating'][a_id]) < int(rating):
            rating = int(left['Rating'][a_id])

    for w_id in cluster.wids:
        if len(right['Name'][w_id]) > len(max_name):
            max_name = right['Name'][w_id]
        if np.isnan(right['Sale Price'][w_id]) == False and
        float(right['Sale Price'][w_id]) > float(max_price):
            max_price = float(right['Sale Price'][w_id])
        if len(right['Author'][w_id]) > len(max_author):
```

```
        max_author = right['Author'][w_id]
    if len(isbn) == 0:
        isbn = right['ISBN10'][w_id]
    if np.isnan(right['Pages'][w_id]) == False and
    int(right['Pages'][w_id]) > int(max_pages):
        max_pages = int(right['Pages'][w_id])
    if len(publisher) == 0:
        publisher = right['Publisher'][w_id]
    if len(language) == 0:
        language = right['Language'][w_id]
    if len(dimensions) == 0 :
        dimensions = right['Dimensions'][w_id]
    if np.isnan(right['Rating'][w_id]) == False and
    int(right['Rating'][w_id]) < int(rating):
        rating = int(right['Rating'][w_id])
if rating == 100:
    rating = math.nan
if max_price == 0:
    max_price = math.nan
if max_pages == 0:
    max_pages = math.nan
writer.writerow([max_name, max_price, category, max_author, isbn,
                 max_pages, publisher, language, dimensions,
                 weight, rating])
```