Data Science - Project Stage 4

Data Matching and Analysis

Team Members:

- Karan Dharni (dharni@wisc.edu)
- Sudarshan Avish Maru (smaru@wisc.edu)
- Amogh Joshi (asjoshi4@wisc.edu)

Merging step:

Table A =Data obtained from www.amazon.com
Table B = Data obtained from www.booksamillion.com

We used the matching tuples from A and B to create table E. We did not add any other data. So, the schema for table E was identical to tables A and B.

Schema: **<ID, Name, Category, Author, Price, Series, Pages, Publisher, Date, Language, ISBN_10, ISBN_13, Dimensions, Weight>**

From our matcher in stage 3, we created a file which contains metadata about the matches. Each line of this file is of the form <lid, rid> which means that tuple with ID=lid in A, matches with tuple with ID=rid in B.
We applied the following merging rules for a value of any attribute:

1. If value is missing in both A and B, we populate an empty string.
2. If value is present in A or B, but not both, we use the value which is either in A (if missing in B) or in B (if missing in A).
3. If value is present in both A and B, then we use value from A except for Price and Pages.
    a. For price, we use the minimum of the values present in A and B.
    b. For pages, we use the maximum amongst the values present in A and B.

Table E statistics:

The schema of table E is:
<Name, Category, Author, Price, Series, Pages, Publisher, Date, Language, ISBN_10, ISBN_13, Dimensions, Weight>

The **number of tuples in table E is 702.**

We had approximately 1000 tuples with actual matches, but we only used the tuples reported as a match by the matcher.

Sample tuples from table E:

Schema:

Name,Category,Author,Price,Series,Pages,Publisher,Date,Language,ISBN_10,ISBN_13,Dimensions,Weight

Sample 1:

Treason,Science Fiction : Space Opera,Gun Brooke,$16.95,,,Bold Strokes Books,"January 15, 2019",English,1635552443,978-1635552447,,1.1 pounds

Sample 2:

Tales from Plexis (Trade Pact Universe),Science Fiction : Space Opera,Julie E. Czerneda,$10.57,Trade Pact Universe,320 pages,DAW,"December 4, 2018",English,0756413931,978-0756413934,5.1 x 8 inches,13 ounces

Sample 3:

"The Emperor's Gift (Warhammer 40,000)",Science Fiction : Space Opera,Aaron Dembski-Bowden,$12.16,"Warhammer 40,000",416 pages,Games Workshop; Reprint edition,"December 11, 2018",English,1784968447,978-1784968441,5.1 x 7.8 inches,1.1 pounds

Sample4:

A Star Wheeled Sky,Science Fiction : Space Opera,Brad R. Torgersen,$12.80,,352 pages,Baen,"December 4, 2018",English,1481483625,978-1481483629,6.1 x 9.8 inches,1.1 pounds

Sample 5:

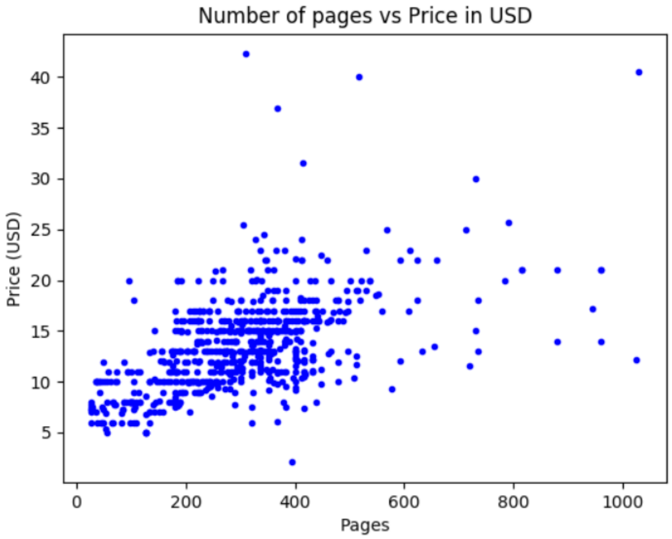
The Magnificent Wilf,Science Fiction : Space Opera,Gordon R. Dickson,$16.00,,272 pages,Baen; Reprint edition,"December 4, 2018",English,1481483633,978-1481483636,6.1 x 9.2 inches,1.1 pounds

Python script for merging: Code appended at the end.

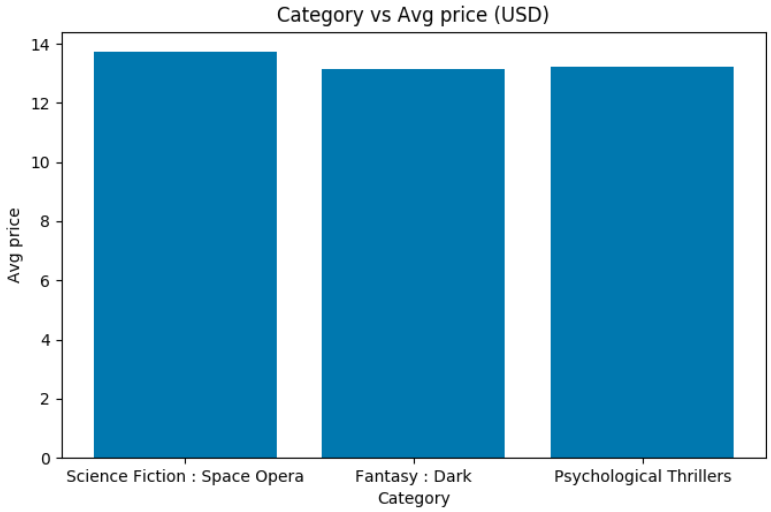
Data Analysis:

We chose **OLAP style exploration** for data analysis. Using various attributes extracted for each book, we tried to find interesting insights using **grouping** and **aggregation** methods. We analyzed the data using the following different ways:

1. Relation between number of pages and price: Our aim was to find if any correlation exists between number of pages in a book and its price. While not completely strict, we did notice a trend that up to a certain point, the price increases with increase in number of pages but further, it almost remains constant.

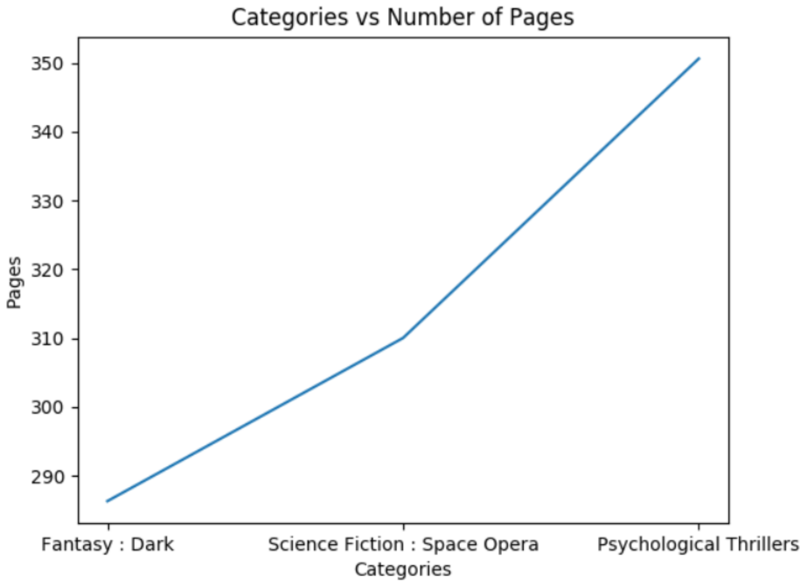


2. Relationship between book category and price: In our dataset (from stage 2), books were limited to 3 categories: Science Fiction: Space Opera, Fantasy: Dark and Psychological Thrillers. We calculated the average price of book for each category. Following are the observations:



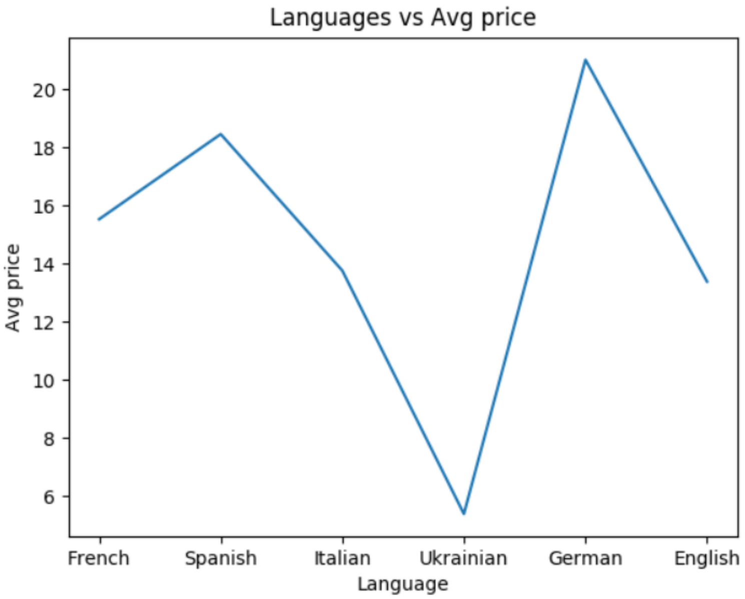
From the graph, we can infer that the average prices of books in all three categories is very similar.

3. Relationship between book category and number of pages: For this part, we calculated the average number of pages in a book of each category. Following are the observations:



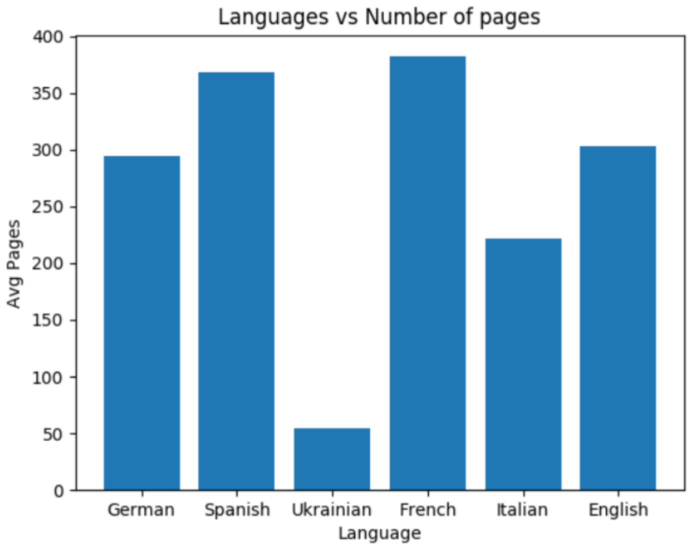
Books in the Psychological Thrillers category have more pages than other 2 categories. Books in the category Fantasy: Dark have the lowest average page size.

4.Relationship between language and price: In our dataset, books written in the following 8 languages were present: **Italian, Spanish, German, English, Ukrainian, French.** We calculated the average price for books published in each language. Following are the insights:



We have very few samples of Ukrainian books, so we cannot comment on its book prices. We have a reasonable sample size for books in French, Spanish, English and German. Books in German particularly are costlier than others. There are many Books in English, but still the average price is quite lower.

5. Relationship between language and number of pages: We calculated the average number of pages for books published in a language. Following are the insights:



In this analysis, we can infer that books in French and Spanish are bigger in size than others.

Problems in Data Analysis:

In general, we did not have any problems due to bad data as it was already handled while creating table E. But while doing analysis, we had a few constraints due to the distribution of the data. There were very few books with a certain attribute (e.g. Books written in Ukrainian were very few, so we could not infer any meaningful insights from such data).

Conclusion from data analysis:

Based on our analysis, we could infer the following:

1. **Relation between number of pages and price:** Up to a certain point, the price increases with increase in number of pages, but further, it almost remains constant.
2. **Relationship between book category and number of pages:** Books in each category vary quite a lot in size. Dark have the lowest average page size.
3. **Relationship between book category and price:** Average prices of books in all three categories are very similar.
4. **Relationship between language and number of pages:** Books in German and English have similar sizes. We do not have sufficient data for books in other

languages. So, we cannot derive any insight based on these few number of samples as it would be incorrect.

5. **Relationship between language and price:** Books in German particularly are costlier than others.

Future Work:

1. In addition to this data, we can scrape average ratings for all books. Then, using ML classification, we can learn a model which can predict average rating of a new book.
2. In this dataset, most of the books are in German, English. We can add more data for books in other languages, so that we can do further analysis.

Python code to merge two tables A and B:

```python
import pandas as pd

COMMA = ","
NEWLINE = "\n"
EMPTY_STRING = ""
SPACE = " "
HEADER = ""
CURRENCY = "$"
PAGES = "pages"
NAN = "nan"

A = pd.read_csv("Data/books_amazon_clean.csv")
B = pd.read_csv("Data/books_millions_clean.csv")
M = pd.read_csv("Data/matches.csv")

A = A.as_matrix()
B = B.as_matrix()
M = M.as_matrix()

def is_empty_or_nan(value):
  if(not str(value) or str(value) == NAN):
     return True
  return False

def merge_matching_books():
  merged_file = open("Data/merged_data.csv", "w")
  merged_file.write(HEADER)
  for i in range(0, len(M)):
```

```
merged_book = EMPTY_STRING
aid = M[i][0]
bid = M[i][1]

for j in range(1, len(A[aid-1])):
    if(is_empty_or_nan(A[aid-1][j]) and is_empty_or_nan((B[bid-1][j]))):
        merged_book += EMPTY_STRING
        merged_book += COMMA
    elif(is_empty_or_nan(A[aid-1][j])):
        merged_book += str(B[bid-1][j])
        if(j == 6):
            merged_book += SPACE
            merged_book += PAGES
        merged_book += COMMA
    elif(is_empty_or_nan(B[bid-1][j])):
        merged_book += str(A[aid-1][j])
        merged_book += COMMA
    elif(j == 4):
        price = str(A[aid-1][j])
        price_amazon = str(A[aid-1][j])
        price_amazon = price_amazon.replace(CURRENCY, EMPTY_STRING)
        price_amazon = price_amazon.strip()
        price_millions = str(B[bid-1][j])
        price_millions = price_millions.replace(CURRENCY, EMPTY_STRING)
        price_millions = price_millions.strip()
        if(float(price_amazon) > float(price_millions)):
            price = str(B[bid-1][j])
        merged_book += price
        merged_book += COMMA
    elif(j == 6):
        pages = str(A[aid - 1][j])
        pages_amazon = str(A[aid - 1][j])
        pages_amazon = pages_amazon.replace(PAGES, EMPTY_STRING)
        pages_amazon = pages_amazon.strip()
        pages_millions = str(B[bid - 1][j])
        pages_millions = pages_millions.strip()
        if (int(pages_amazon) < int(round(float(pages_millions)))):
            pages = str(int(round(float(pages_millions))))
            pages += SPACE
            pages += PAGES
        merged_book += pages
        merged_book += COMMA
    else:
        merged_book += str(A[aid-1][j])
        merged_book += COMMA

merged_book = merged_book[0:len(merged_book)-1]
merged_book += NEWLINE
merged_file.write(merged_book)
```

```
    merged_file.close()

def set_header():
    global HEADER
    amazon_file = open("Data/books_amazon_clean.csv", "r")
    HEADER = amazon_file.readline()
    amazon_file.close()

    first_comma_index = -1
    for i in range(0, len(HEADER)):
        if(HEADER[i] == ','):
            first_comma_index = i
            break;
    HEADER = HEADER[first_comma_index+1:len(HEADER)]


set_header()
merge_matching_books()
```