Algorithms for massive data

«Link analysis»

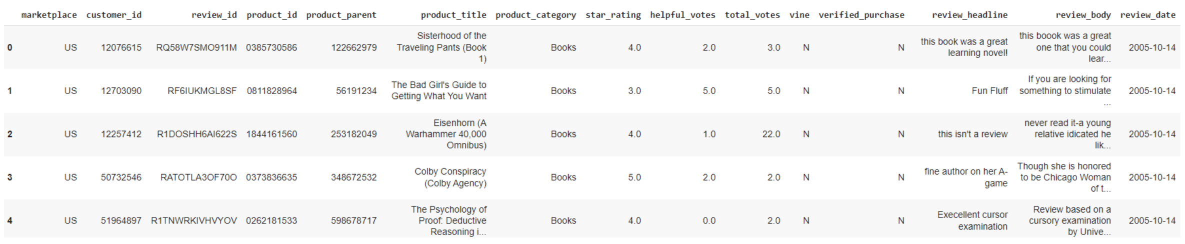
Author: Soboleva Daria

July 18, 2023

1 Dataset

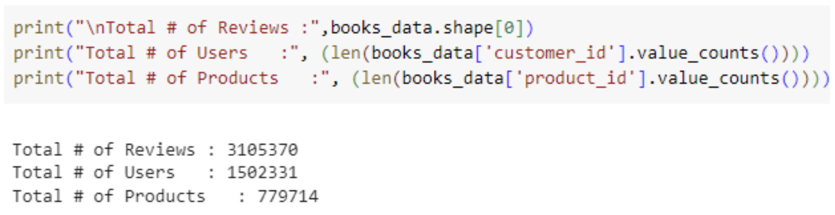
In my analysis, I used the ≪Amazon US Customer Review≫ dataset, published on Kaggle under the amazon.com conditions of use. The selected category is “Books”. Originally dataset consists of 15 columns:

`{marketplace, customer_id, review_id, product_id, product_parent, product_title, product_category, star_rating, helpful_votes, total_votes, vine, verified_purchase, review_headline, review_body, review_date}`



	marketplace	customer_id	review_id	product_id	product_parent	product_title	product_category	star_rating	helpful_votes	total_votes	vine	verified_purchase	review_headline	review_body	review_date
0	US	12076615	RQ58W7SMO911M	0385730586	122662979	Sisterhood of the Traveling Pants (Book 1)	Books	4.0	2.0	3.0	N	N	this book was a great learning novel!	this boook was a great one that you could lear...	2005-10-14
1	US	12703090	RF6IUKMGL8SF	0811828964	56191234	The Bad Girl's Guide to Getting What You Want	Books	3.0	5.0	5.0	N	N	Fun Fluff	If you are looking for something to stimulate ...	2005-10-14
2	US	12257412	R1DOSHH6AI622S	1844161560	253182049	Eisenhorn (A Warhammer 40,000 Omnibus)	Books	4.0	1.0	22.0	N	N	this isn't a review	never read it-a young relative idicated he lik...	2005-10-14
3	US	50732546	RATOTLA3OF70O	0373836635	348672532	Colby Conspiracy (Colby Agency)	Books	5.0	2.0	2.0	N	N	fine author on her A-game	Though she is honored to be Chicago Woman of t...	2005-10-14
4	US	51964897	R1TNWRKIVHVYOV	0262181533	598678717	The Psychology of Proof: Deductive Reasoning i...	Books	4.0	0.0	2.0	N	N	Execellent cursor examination	Review based on a cursory examination by Unive...	2005-10-14

Since I don’t need all variables for my research, I drop 13 columns and keep only customer ID and product ID. The number of records in the chosen category – 3 105 372. Total number of reviews, customers, and products:

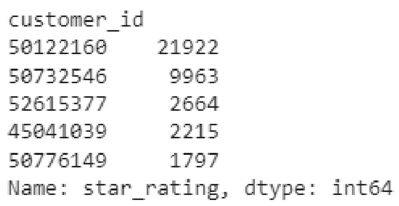


```
print("\nTotal # of Reviews :",books_data.shape[0])
print("Total # of Users   :", (len(books_data['customer_id'].value_counts())))
print("Total # of Products   :", (len(books_data['product_id'].value_counts())))
```

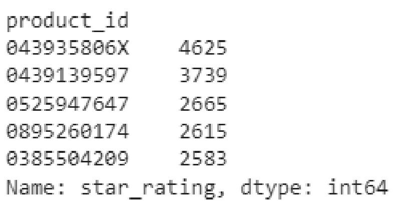
```
Total # of Reviews : 3105370
Total # of Users   : 1502331
Total # of Products   : 779714
```

2 Data analysis

The maximum number of reviews was made by customer 50122160. The product which received the maximum number of reviews is 043935806X (“Harry Potter and the Order of the Phoenix (Book 5)”).

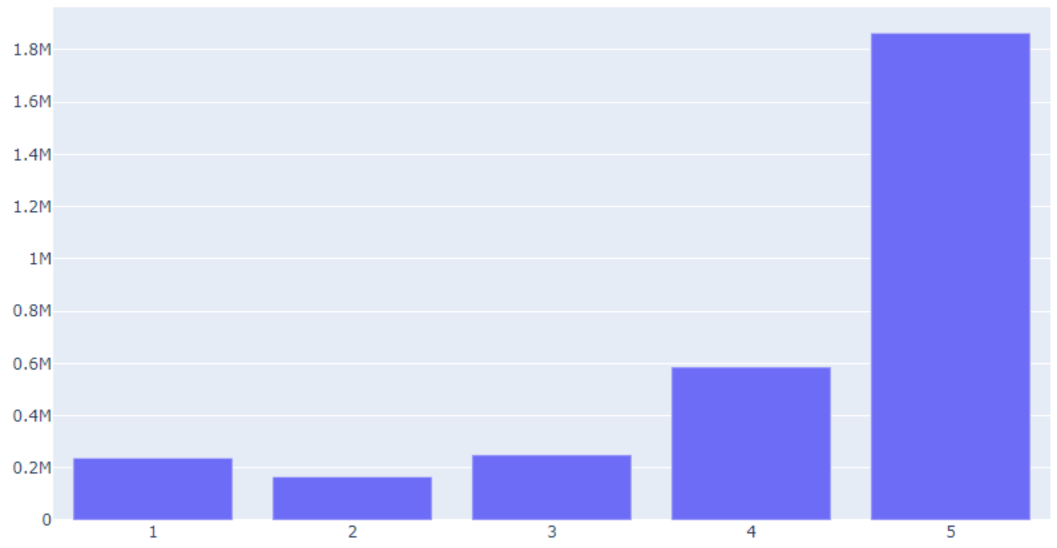


```
customer_id
50122160    21922
50732546     9963
52615377     2664
45041039     2215
50776149     1797
Name: star_rating, dtype: int64
```



```
product_id
043935806X    4625
0439139597    3739
0525947647    2665
0895260174    2615
0385504209    2583
Name: star_rating, dtype: int64
```

Ratings distribution:



Most of the reviews are positive: 1.86 million reviews are “5 stars”, and 0.586 million are “4 stars”.

3 PageRank Algorithm

PageRank is a web page ranking algorithm developed in 1996 by founders of Google: Larry Page and Sergey Brin. According to PageRank important pages are likely to receive more links from other pages. It is a kind of ”voting” pages for each other. However, the importance of each page is taken into account based on its own importance and the number of links it gives to other pages. Thus, PageRank considers not only the number of links, but also their quality. The algorithm works by assigning a numerical weight, called the PageRank score, to each page in the graph. The PageRank score represents the probability that a random surfer, following links on the web, will land on a particular page.

The PageRank score for a page is calculated by the following formula:

$$PR(page) = (1 - d) + d \cdot \sum_i \left(\frac{PR(t_i)}{C(t_i)} \right)$$

where:

- $PR(page)$ is the PageRank score of the page
- d is the damping factor: probability that a random surfer will continue clicking on links rather than jumping to a random page (typically set to 0.85)
- $PR(t_i)$ is the PageRank score of a page t_i that links to the current page
- $C(t_i)$ is the total number of outgoing links on page t_i

4 Algorithm implementation

The goal of this project is to implement a ranking system based on the PageRank index to Amazon reviews dataset. The entities could be ranked either customers (link between two customers if they have reviewed at least a same product) or products (two products will be linked if they have been reviewed at least by a same customer).

For both cases necessary steps are:

1. Create an empty graph
2. Add edges between customers who reviewed the same product or between products reviewed by the same customer
3. Calculate PageRank
4. Sort customers/products by PageRank

4.1 Customer-based Ranking System

The dataset contains information about reviews of 1,502,331 customers. I faced a problem trying to execute the PageRank algorithm on the whole dataset. Therefore, to reduce computational complexity I will be focused on Top N customers (who gave the most reviews).

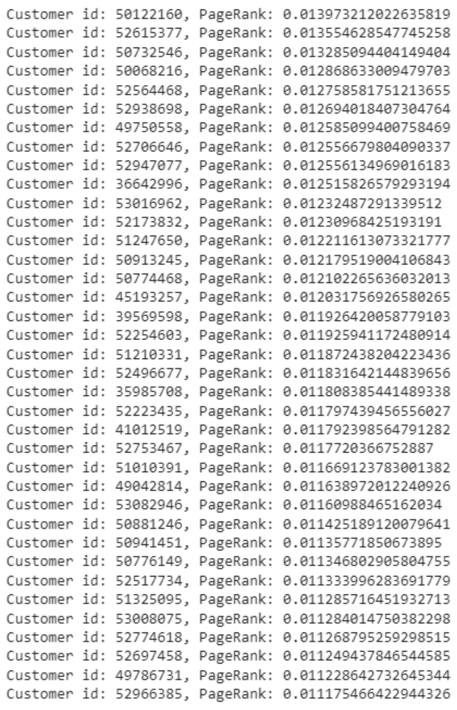
```
customer_review_counts = books_df['customer_id'].value_counts()
top_n_customers = N
top_customers = customer_review_counts.head(top_n_customers).index.tolist()
filtered_df = books_df[books_df['customer_id'].isin(top_customers)]
```

Listing 1: Pre-processing

In general, the algorithm is:

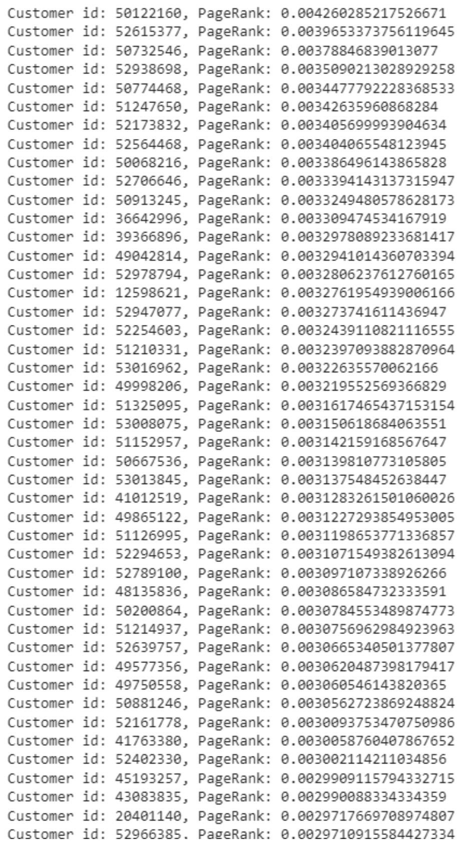
```
G = nx.Graph()
for _, row in filtered_df.iterrows():
    product_id = row['product_id']
    customers = filtered_df[filtered_df['product_id'] == product_id]['
        customer_id'].values
    for i in range(len(customers)):
        for j in range(i + 1, len(customers)):
            G.add_edge(customers[i], customers[j])
page_rank = nx.pagerank(G)
sorted_customers = sorted(page_rank, key=page_rank.get, reverse=True)
for customer_id in sorted_customers:
    page_rank_value = page_rank[customer_id]
    print(f"Customer id: {customer_id}, PageRank: {page_rank_value}")
```

I have implemented this algorithm for N=10, N=100 and N=500. In Top 500 there are customers who reviewed more than 190 products, in Top 100 - users who gave more than 470 reviews. Below you can see the results given for N=100 and N=500.



```
Customer id: 50122160, PageRank: 0.013973212022635819
Customer id: 52615377, PageRank: 0.013554628547745258
Customer id: 50732546, PageRank: 0.013285094404149404
Customer id: 50068216, PageRank: 0.012868633009479703
Customer id: 52564468, PageRank: 0.012758581751213655
Customer id: 52938698, PageRank: 0.012694018407304764
Customer id: 49750558, PageRank: 0.012585099400758469
Customer id: 52706646, PageRank: 0.012556679804090337
Customer id: 52947077, PageRank: 0.012556134969016183
Customer id: 36642996, PageRank: 0.012515826579293194
Customer id: 53016962, PageRank: 0.01232487291339512
Customer id: 52173832, PageRank: 0.01230968425193191
Customer id: 51247650, PageRank: 0.012211613073321777
Customer id: 50913245, PageRank: 0.012179519004106843
Customer id: 50774468, PageRank: 0.012102265636032013
Customer id: 45193257, PageRank: 0.012031756926580265
Customer id: 39569598, PageRank: 0.011926420058779103
Customer id: 52254603, PageRank: 0.011925941172480914
Customer id: 51210331, PageRank: 0.011872438204223436
Customer id: 52496677, PageRank: 0.011831642144839656
Customer id: 35985708, PageRank: 0.011808385441489338
Customer id: 52223435, PageRank: 0.011797439456556027
Customer id: 41012519, PageRank: 0.011792398564791282
Customer id: 52753467, PageRank: 0.011772036675288 7
Customer id: 51010391, PageRank: 0.011669123783001382
Customer id: 49042814, PageRank: 0.011638972012240926
Customer id: 53082946, PageRank: 0.01160988465162034
Customer id: 50881246, PageRank: 0.011425189120079641
Customer id: 50941451, PageRank: 0.01135771850673895
Customer id: 50776149, PageRank: 0.011346802905804755
Customer id: 52517734, PageRank: 0.011333996283691779
Customer id: 51325095, PageRank: 0.011285716451932713
Customer id: 53008075, PageRank: 0.011284014750382298
Customer id: 52774618, PageRank: 0.011268795259298515
Customer id: 52697458, PageRank: 0.011249437846544585
Customer id: 49786731, PageRank: 0.011228642732645344
Customer id: 52966385, PageRank: 0.011175466422944326
```

[N=100]



```
Customer id: 50122160, PageRank: 0.004260285217526671
Customer id: 52615377, PageRank: 0.0039653373756119645
Customer id: 50732546, PageRank: 0.00378846839013077
Customer id: 52938698, PageRank: 0.0035090213028929258
Customer id: 50774468, PageRank: 0.0034477792228368533
Customer id: 51247650, PageRank: 0.00342635960868284
Customer id: 52173832, PageRank: 0.003405699993904634
Customer id: 52564468, PageRank: 0.003404065548123945
Customer id: 50068216, PageRank: 0.003386496143865828
Customer id: 52706646, PageRank: 0.0033394143137315947
Customer id: 50913245, PageRank: 0.0033249480578628173
Customer id: 36642996, PageRank: 0.003309474534167919
Customer id: 39366896, PageRank: 0.0032978089233681417
Customer id: 49042814, PageRank: 0.0032941014360703394
Customer id: 52978794, PageRank: 0.0032806237612760165
Customer id: 12598621, PageRank: 0.0032761954939006166
Customer id: 52947077, PageRank: 0.003273741611436947
Customer id: 52254603, PageRank: 0.0032439110821116555
Customer id: 51210331, PageRank: 0.0032397093882870964
Customer id: 53016962, PageRank: 0.00322635570062166
Customer id: 49998206, PageRank: 0.003219552569366829
Customer id: 51325095, PageRank: 0.0031617465437153154
Customer id: 53008075, PageRank: 0.003150618684063551
Customer id: 51152957, PageRank: 0.003142159168567647
Customer id: 50667536, PageRank: 0.003139810773105805
Customer id: 53013845, PageRank: 0.003137548452638447
Customer id: 41012519, PageRank: 0.0031283261501060026
Customer id: 49865122, PageRank: 0.0031227293854953005
Customer id: 51126995, PageRank: 0.0031198653771336857
Customer id: 52294653, PageRank: 0.0031071549382613094
Customer id: 52789100, PageRank: 0.003097107338926266
Customer id: 48135836, PageRank: 0.003086584732333591
Customer id: 50200864, PageRank: 0.0030784553489874773
Customer id: 51214937, PageRank: 0.0030756962984923963
Customer id: 52639757, PageRank: 0.0030665340501377807
Customer id: 49577356, PageRank: 0.0030620487398179417
Customer id: 49750558, PageRank: 0.003060546143820365
Customer id: 50881246, PageRank: 0.0030562723869248824
Customer id: 52161778, PageRank: 0.0030093753470750986
Customer id: 41763380, PageRank: 0.0030058760407867652
Customer id: 52402330, PageRank: 0.003002114211034856
Customer id: 45193257, PageRank: 0.0029909115794332715
Customer id: 43083835, PageRank: 0.002990088334334359
Customer id: 20401140, PageRank: 0.0029717669708974807
Customer id: 52966385. PageRank: 0.0029710915584427334
```

[N=500]

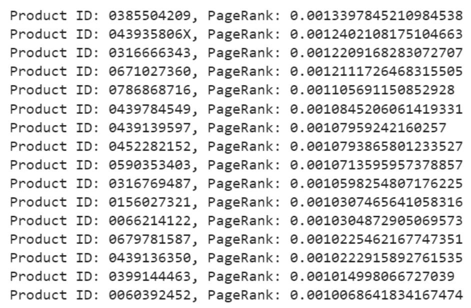
According to the results, the most influential customers are 50122160, 52615377, 50732546 (they have the highest PageRank values). In case of N=500 probabilities are very low, and obviously, reducing number of examined nodes, probabilities will increase (as we can see for N=100).

4.2 Product-based Ranking System

In the dataset there are 779,714 products and 3,105,370 reviews. To have more reliable data I take into consideration only reviews from customers who have made more than 10 reviews and reviews for products that have received more than 15 reviews. PageRank algorithm will be:

```python
G_new = nx.Graph()
for _, row in f_df.iterrows():
    customer_id = row['customer_id']
    products = f_df[f_df['customer_id'] == customer_id]['product_id'].values
    for i in range(len(products)):
        for j in range(i + 1, len(products)):
            G_new.add_edge(products[i], products[j])
page_rank = nx.pagerank(G_new)
sorted_products = sorted(page_rank, key=page_rank.get, reverse=True)
for product_id in sorted_products:
    page_rank_value = page_rank[product_id]
    print(f"Product ID: {product_id}, PageRank: {page_rank_value}")
```

Results of implementing PageRank algorithm:



```
Product ID: 0385504209, PageRank: 0.0013397845210984538
Product ID: 043935806X, PageRank: 0.0012402108175104663
Product ID: 0316666343, PageRank: 0.0012209168283072707
Product ID: 0671027360, PageRank: 0.0012111726468315505
Product ID: 0786868716, PageRank: 0.001105691150852928
Product ID: 0439784549, PageRank: 0.0010845206061419331
Product ID: 0439139597, PageRank: 0.00107959242160257
Product ID: 0452282152, PageRank: 0.0010793865801233527
Product ID: 0590353403, PageRank: 0.0010713595957378857
Product ID: 0316769487, PageRank: 0.0010598254807176225
Product ID: 0156027321, PageRank: 0.0010307465641058316
Product ID: 0066214122, PageRank: 0.0010304872905069573
Product ID: 0679781587, PageRank: 0.0010225462167747351
Product ID: 0439136350, PageRank: 0.0010222915892761535
Product ID: 0399144463, PageRank: 0.001014998066727039
Product ID: 0060392452, PageRank: 0.0010068641834167474
```

The most linked products in category ”Books”: 0385504209 (The Da Vinci Code), 043935806X (Harry Potter and the Order of the Phoenix (Book 5)), 0316666343 (The Lovely Bones).

5 Declaration

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.