Algorithms for massive data (DSE)

Link Analysis of the Amazon Book Review Dataset

December 13, 2025

1. Dataset Description

The dataset used in this study is the Amazon Books Reviews dataset, publicly available on Kaggle. It contains 3 million reviews for a wide variety of books and includes multiple columns:

- Id: Unique identifier for each book.
- Title: The book's title.
- Price: The price of the book.
- User_id: Unique identifier for each user.
- profileName: User profile name.
- review/helpfulness: Metrics of review helpfulness.
- review/score: Review score, ranging from 1 to 5.
- review/time: Review timestamp.
- review/summary and text: Content of the review.

For this project, a subset of columns was considered to simplify analysis and focus on network interactions: - Id → renamed to book_id - UserId → renamed to user_id - Score

To enable iterative experimentation, a subsampling parameter (USE_SUBSAMPLE = True) was applied to take a small fraction (default 5%) of the dataset, without altering the underlying graph relationships.

2. Data Organization

The analysis focuses on **graph-based representations** of user and book interactions. Two main graphs were constructed:

1. **User–User Graph**
    - **Nodes:** Users
    - **Edges:** An edge exists between two users if they have both reviewed at least one common book.
    - This represents shared interests and potential influence in the review community.
2. **Book–Book Graph**
    - **Nodes:** Books

- **Edges:** An edge exists between two books if at least two users have reviewed both.
- This captures implicit connections between books through shared readership and allows identifying structurally important books beyond sheer popularity.

Data was represented as PySpark DataFrames, allowing distributed computation and efficient handling of large datasets. Edges were represented as pairs of src and dst nodes, and vertices were derived from the union of all unique src and dst entries.

3. Pre-processing Techniques

The preprocessing workflow included several key steps:

1. **Column Selection and Renaming:**
   - Retained only book_id, user_id, and score.
2. **Handling Missing Data:**
   - Dropped rows where either book_id or user_id was null.
3. **Filtering Users and Books:**
   - Users with fewer than three reviews were excluded.
   - Books with fewer than five reviews were excluded.
   - This ensures that the graph nodes are active and reduces sparsity.
4. **Graph Construction:**
   - **User–User:** Collected all users per book, exploded pairs, removed self-loops, and deduplicated.
   - **Book–Book:** Collected all books per user, exploded pairs, counted shared users per book pair, filtered to pairs with ≥2 shared users.

These steps ensured that the graphs represented meaningful connections and were ready for PageRank computation.

4. Algorithms and Implementations

4.1 PageRank Theory

PageRank is an algorithm that assigns a **centrality score** to nodes in a graph based on the principle that a node is important if many important nodes point to it. The iterative formula is:

Mathematically, the PageRank score of a node P is:

$$PageRank(P) = (1 - d)/N + d \sum \frac{PageRank(Q)}{L(Q)}$$

Where:

- `d`: Damping factor (default `0.85`), representing the probability of following a link versus jumping to a random node.
- `N`: Total number of nodes.
- `Σ (PageRank(Q) / L(Q))`: Sum of PageRank scores from all linking nodes `Q`, divided by their outgoing links `L(Q)`.

The algorithm converges iteratively, distributing rank scores across the graph until changes are below a threshold or a fixed number of iterations is reached.

4.2 Custom PySpark DataFrame Implementation

- **Vertices:** Extracted as the union of src and dst columns.
- **Initialization:** All nodes set to rank = 1.0.
- **Iteration:** For each edge, the rank of the source node is divided among its out-edges; contributions are summed at the destination node.
- **Damping Factor:** Applied to account for random teleportation.

Advantages: - Fully Spark-native. - No external dependencies beyond PySpark. - Scales with the number of edges.

4.3 GraphFrames Implementation

- Leverages GraphFrame.pageRank(resetProbability=0.15, maxIter=10).
- Internally optimized for distributed computation.
- Provides vertex-level pagerank column directly.
- Suitable for very large graphs and production-scale datasets.

5. Scalability Considerations

The proposed solution scales efficiently because:

1. **Spark DataFrames** enable distributed computation of edges and iterative updates.
2. **Graph construction** relies on groupBy and explode operations, which scale linearly with the number of rows.
3. **PageRank iterations** involve joins and aggregations; these are parallelized across the Spark cluster.
4. **Subsampling** allows quick testing without changing logic, maintaining scalability for the full dataset.
5. **GraphFrames** offers built-in optimizations for large-scale PageRank computation.

Even with millions of reviews, the workflow can run on modest hardware when subsampling, and scale linearly for full dataset execution.

6. Experiments

6.1 Graph Validation

- **User–User Graph:** Nodes = active users; edges = shared book reviews.

```
Top Users
+--------------+---------------------+
|id            |rank                 |
+--------------+---------------------+
|A1D2C0WDCSHUWZ|7.592234690615566E23 |
|A35PU6DDT4SJXK|6.308519027007406E23 |
|A259F4J3QOBIUH|5.794680330627896E23 |
|A311QBN2U5AIQN|4.304149118238791E23 |
|AWZ8FK8JGPYYG |3.9783595025635274E23|
|AVHBK2SJ6CJ1C |3.9664771526986045E23|
|AAIL33CYCT47J |3.8389472823323436E23|
|A2U21BHZJONNGD|3.833571640746973E23 |
|A2E5IE7HKSQJ7E|3.825675983338078E23 |
|A22GETHK36NV30|3.756503085013428E23 |
+--------------+---------------------+
```

- **Book–Book Graph:** Nodes = popular books; edges = ≥2 shared reviewers.

Top Books with Titles

book_id	title	rank
B000EVI8O0	Pride and Prejudice	1.8425225630048605E11
1901768945	Pride and Prejudice	1.8235574079991473E11
B000F6H01Q	Pride and Prejudice	1.625173707646191E11
1844560333	Pride and Prejudice	1.6057837443144934E11
B000GDLGSG	Pride and Prejudice	1.569224128118992E11
0786135034	Pride and Prejudice	1.424604156252615E11
B000C1X8JC	Pride and Prejudice	1.3649020465968379E11
9562911306	Pride and Prejudice	1.3337505631895695E11
1847022251	Jane Eyre (Large Print)	1.2368225890196198E11
0451513967	Pride and Prejudice	1.1905030380509755E11
0435126075	Pride & Prejudice (New Windmill)	1.1521165005620302E11
0195813618	Emma (Progress English)	1.1296430195723851E11
B000KORO82	Little Women or Meg, Jo, Beth, and Amy.	1.0884471163882152E11
1581730470	Emma	1.0864987430284764E11
B0006F3T8Y	Little women, or, Meg, Jo, Beth, and Amy	1.0807248943772711E11
B0007EGQ52	Little women: Or Meg, Jo, Beth, and Amy	1.0081052585419475E11
0395051150	Emma (Riverside Editions)	1.0027962737473041E11
0460112872	Jane Eyre (Everyman's Classics)	9.889446316722984E10
0141804459	Pride & Prejudice (Penguin Classics)	9.845389528475685E10
8188280038	Sense & Sensibility (Classic Library)	9.807716347746062E10

6.2 PageRank Computation

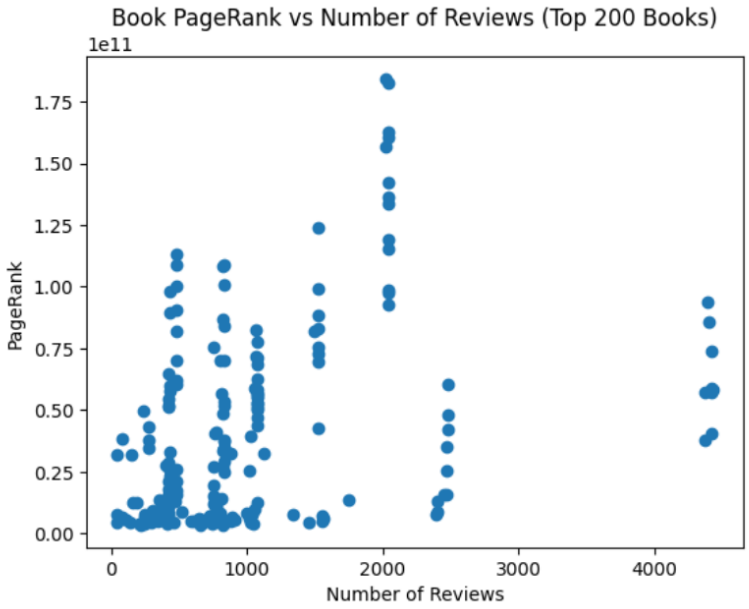
- Both custom DataFrame and GraphFrames implementations were applied.
- Iterations: 10; damping factor: 0.85 (custom) or resetProbability 0.15 (GraphFrames).

6.3 Top-Ranked Entities

- **Users:** high PageRank indicates influential users, connected to other active reviewers.
- **Books:** high PageRank indicates books connected via influential users.
- Top 10 results from both implementations were compared and consistent.

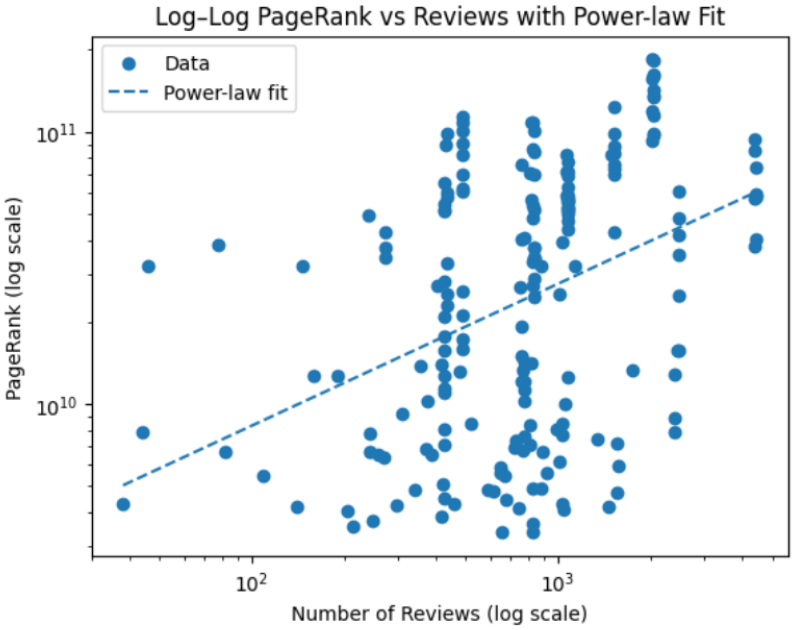
6.4 Further Analysis

- Plotted PageRank vs number of reviews to assess correlation.
-



- Fitted a power-law trend line using $\log(PR) = \alpha \log(NumReviews) + \beta$ to assess to assess structural influence beyond popularity.

```
Power-law fit: PageRank ≈ exp(20.446) * reviews^0.521
```



7. Comments and Discussion

- **PageRank vs raw review count:** Many books with fewer reviews but strong connectivity achieved higher PageRank.
- **User influence:** Some users with moderate review counts but connections to other active users ranked higher.
- **Algorithm comparison:** Custom DataFrame and GraphFrames implementations yielded nearly identical rankings, confirming correctness.
- **Interpretability:** Joining book titles to PageRank scores enables meaningful presentation.
- **Network insights:** The book–book graph reveals structural relationships not captured by review count alone.
- **Scalability:** Subsampling permits rapid experimentation; full dataset computation is feasible with distributed Spark.

In conclusion, applying PageRank to both users and books allows identification of central entities in the network, capturing influence and structural importance beyond simple metrics. The combination of Spark DataFrames and GraphFrames ensures that the approach is scalable, reproducible, and suitable for large-scale review datasets.

"I/We declare that this material, which I/We now submit for assessment, is entirely my/our 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/our work, and including any code produced using generative AI systems. I/We 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/us or any other person for assessment on this or any other course of study."