Scalable Clustering of Amazon Book Reviews using the BFR Algorithm

Alex Tulip

Department of Computer Science, University of Milan, Italy.

Abstract

In this report, I present a scalable clustering system that I implemented from scratch to analyze the Amazon Books Review dataset. Because the dataset is massive (3GB+), I used the Bradley, Fayyad, and Reina (BFR) algorithm, which is a version of K-Means designed for data that sits on a hard disk rather than in RAM. My system processes the data in chunks, keeping track of statistical summaries ($\boldsymbol{N, SUM, SUMSQ}$) so it can cluster everything in just one pass. My analysis found $\boldsymbol{K = 4}$ distinct market segments in the book industry, distinguishing products not just by price, but by an interesting non-linear relationship between how popular they are and how highly they are rated.

1 Introduction and Dataset

The goal of this project is to find hidden structures in the book market using unsupervised learning techniques that can handle massive datasets.

I based my analysis on the **Amazon Books Review** dataset hosted on Kaggle. The raw dataset has over 3 million reviews and is about 3GB in size. Because of memory limits, it was impossible to load the whole thing into RAM at once. So, I accessed the data using a streaming approach, reading the CSV file in chunks of 100,000 rows at a time.

1.1 Data Organization

The raw data consists of individual reviews. To do meaningful clustering, I aggregated these reviews to create a profile for each unique book. I used a dictionary-based hash map to stream the data and update the following statistics for each book title:

- **Count:** The total number of reviews.

- **Sum of Ratings:** Used to calculate the average rating.
- **Price Information:** Used to calculate the average price (this also helped handle some inconsistencies in the metadata).

This step reduced the dataset from millions of review rows to about 48,000 unique book entities, which served as the data points for my clustering algorithm.

2 Preprocessing

The BFR algorithm works in a Euclidean space and uses the Mahalanobis distance. For this to work, the features need to be numerical and comparable. I chose three features for clustering:

1. **Price:** How much the book costs.
2. **Average Rating:** A score between 1 and 5.
3. **Review Count:** A proxy for how popular the book is.

2.1 Normalization

These features have very different scales (e.g., Price is usually around 20, Rating is around 4, but Count can be in the thousands). To stop the "Review Count" from dominating the distance calculations, I applied **Z-score normalization** to all features:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

This ensures that all dimensions contribute equally to the cluster definition, which is critical for the Mahalanobis distance used in BFR.

3 Algorithms and Implementation

I wrote the code for the **BFR (Bradley, Fayyad, and Reina) Algorithm** entirely from scratch in Python. BFR is a variation of K-Means specifically designed for high-dimensional data that doesn't fit in main memory.

3.1 BFR Architecture

My implementation manages three distinct sets of points, as defined during the course:

- **Discard Set (DS):** Points that are close enough to a cluster centroid to be summarized. I represent these only by their statistics ($N, \mathbf{SUM}, \mathbf{SUMSQ}$) and discard the points themselves.
- **Compressed Set (CS):** Points that are outliers relative to the main clusters but are close to each other. These form "miniclusters" and are also summarized.
- **Retained Set (RS):** True outliers that don't fit into the DS or the CS. I keep these in memory as individual points.

3.2 Mahalanobis Distance and Variance Fix

Points are assigned to the Discard Set if their Mahalanobis distance to a centroid is below a threshold (which I set to 3.0).

$$d(\mathbf{x}, \mathbf{c}) = \sqrt{\sum_{i=1}^{d} \left(\frac{x_i - c_i}{\sigma_i} \right)^2} \tag{2}$$

Implementation Challenge: During initialization, clusters formed with a single point have a variance of zero. This causes the Mahalanobis distance to become undefined (division by zero), so the cluster can't grow. **Solution:** I used a heuristic where clusters with $N < 10$ use a default standard deviation of $\sigma = 1.0$ (matching the normalized global variance). This allows new clusters to "warm up" and absorb their first few neighbors before switching to their actual calculated variance.

4 Scalability

My solution scales linearly $O(N)$ with the size of the dataset. By processing the raw CSV in chunks and strictly maintaining summary statistics, the memory footprint depends on the number of clusters K, not the number of data points N. The aggregation step reduces the data volume significantly, and the BFR logic ensures that even if the number of books increased to millions, the memory usage would remain stable because points are continuously discarded into summary vectors.

5 Experiments

I found the optimal number of clusters K using the Elbow Method, testing values from $K = 2$ to $K = 6$. I ran the full BFR pipeline for each K and calculated the Sum of Squared Errors (SSE) for the Discard Set.

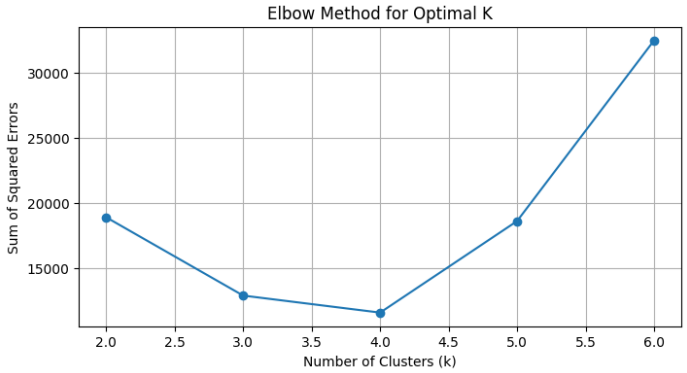

Fig. 1 Elbow Method showing the SSE for different K values. Note the distinct minimum at K=4 and the instability (SSE increase) at K=5 and K=6.

As shown in Fig. 1, the SSE decreases as expected from $K = 2$ to $K = 4$. However, at $K = 5$, the error actually goes up significantly. This "inverted elbow" suggests that forcing more than 4 clusters makes the model unstable, causing the BFR algorithm to expand cluster variances too much to absorb points, or rejecting too many points into the Retained Set. So, I chose $K = 4$ as the best configuration.

6 Results and Discussion

The clustering results for $K = 4$ reveal distinct market segments in the book industry. Fig. 2 visualizes these segments.

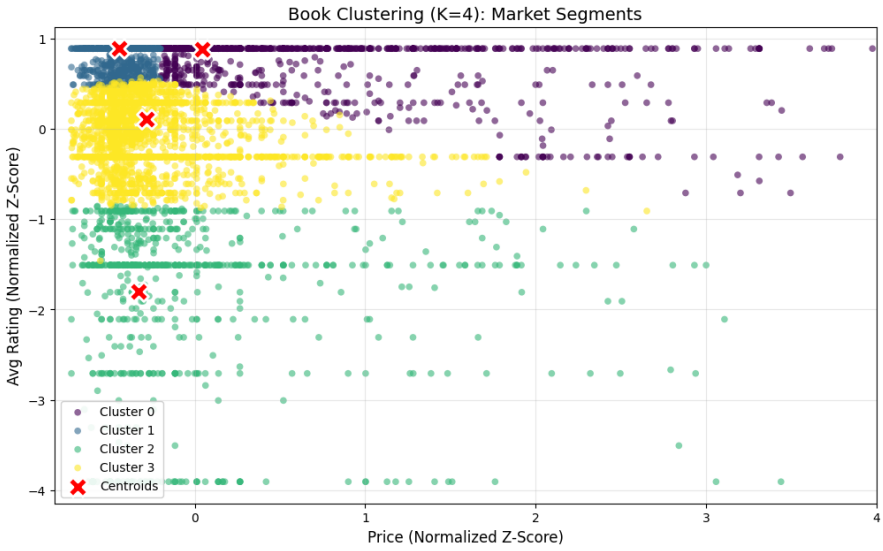

Fig. 2 Visualizing the 4 Market Segments (K=4). The zoomed view focuses on the main density of data, with the centroids marked by red crosses.

I interpreted the four identified clusters as follows:

1. **Cluster 0 (Expensive/Niche):** High price ($\approx$ \$31), perfect rating (5.0), but very low review counts. These are likely niche textbooks or collectors' items.
2. **Cluster 1 (Cheap/Top-Rated):** The largest cluster. Low price ($\approx$ \$12) with 5-star ratings. These are the "hidden gems" of the catalog.
3. **Cluster 2 (The Flops):** A distinct group defined by low ratings ($\approx$ 2.7 stars). BFR successfully separated these underperforming products from the generally high-rated baseline.
4. **Cluster 3 (Mainstream):** Characterized by the highest review counts (avg 7) and solid ratings (4.3 stars). These are popular, widely-read books.

The vertical alignment of centroids in Fig. 2 highlights that for most books (Clusters 1, 2, 3), price isn't the main differentiating factor; the market segments are defined primarily by quality (Rating) and popularity (Review Count).

Declaration of Authorship

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. No generative AI tool has been used to write the code or the report content.