Market Basket Analysis of Amazon Book Reviews
Data Science for Economics

Algorithm For Massive Data

Professor Dario Malchiodi

Araz Asgharieh Ahari - 09786Q

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, and including any code produced using generative AI systems. 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.

1. Introduction

This project focuses on applying Market Basket Analysis techniques to extract meaningful co-occurrence patterns from user reviews of books on Amazon. The objective is to find frequent word combinations that appear in reviews and derive association rules that help understand the structure of review texts.

2. Dataset Description

The dataset used is "Amazon Books Reviews" published on Kaggle by user **mohamedbakhet**. The original dataset contains various fields related to book reviews, such as:

Id
Title
Price
User_id
profileName
review/helpfulness
review/score
review/time
review/summary
review/text

Only the review/text column was considered for this project, as it contains the raw user-generated content used for mining patterns.

3. Data Organization

The dataset was loaded into a PySpark DataFrame using Spark 3.1.1. Null values in the `review/text` field were dropped. A random 10% sample was selected for analysis to reduce memory usage and improve performance in Google Colab.

4. Pre-processing Techniques

The following pre-processing steps were applied to the sampled reviews:

Lowercasing all text

Removing punctuation using regular expressions
Tokenizing the text into individual words (baskets)
Removing duplicate words in each basket using array_distinct
Removing English stopwords using Spark's StopWordsRemover

This produced clean and distinct baskets of non-trivial words per review.

5. Algorithms and Implementation

The core algorithm used is FP-Growth (Frequent Pattern Growth) from PySpark's ml.fpm module. The algorithm was applied with the following parameters:

minSupport = 0.05
minConfidence = 0.1

This model returned:

* Frequent itemsets: frequently co-occurring words
* Association rules: logical rules such as "if word A and B appear, then word C likely appears"

6. Scalability

Spark's distributed framework was used to manage scalability. The model worked effectively with a 10% sample from the full dataset, which included over 1.3 million rows. Reducing the input size and avoiding full memory expansion prevented out-of-memory issues during FP-Growth training.

However, larger-scale analysis can be done by:

* Increasing available memory (Colab Pro or Spark cluster)
* Using batch processing with saved intermediate results

7. Experimental Description

Two main sets of experiments were run:

1. Frequent Itemsets: After preprocessing, top 10 most frequent itemsets were extracted and visualized. The most common words included book, read, story, like, and good, showing their prevalence in reviews.

2. Association Rules: High-confidence rules like:

  * If people and read appear, then book also appears
  * If recommend, then likely book appears

These indicate strong contextual relationships between user sentiment and book references.

8. Discussion

Correctness of Approach

The FP-Growth algorithm is a well-established and correct method for frequent itemset mining. Its application to review token baskets is conceptually sound.

Replicability

All preprocessing steps, sampling, and algorithmic settings were clearly defined. Anyone with the same dataset and Spark setup can replicate results.

Scalability

Using Spark ensured that even a relatively large dataset could be processed efficiently with limited resources. However, scaling beyond 10% may require more robust infrastructure.

Clarity

The steps were modular and interpretable, allowing clear insight into how the words and patterns were extracted, cleaned, and analyzed.

Limitations and Improvements

Token-level analysis loses multi-word expressions (e.g., "science fiction")
Frequent itemsets contain generic terms (e.g., book, read) despite stopword filtering
Further improvements could include:

* Named Entity Recognition to capture book titles
* Fuzzy matching to merge similar terms (e.g., `books`, `book`)
* Phrase modeling using bigrams or trigrams

Note: No pre-existing Kaggle solution was copied. While the dataset is hosted on Kaggle, all code and processing logic were developed independently.