From Raw Data to Informed Decisions: Analyzing Amazon Book Reviews

Alberti A. • Ligari D. •Andreoli C. [1]

[1] *Department of Computer Engineering - Data Science, University of Pavia, Italy*
Course of Data science and big data analytics

Github repository: *https://github.com/DavideLigari01/data-science-project*

Abstract

Keywords— *DNS reflection and amplification Attacks • Amplification factor • Ping • Wireshark • DIG • Mitigation measures*

Contents

1 Introduction

In the age of digital commerce, customer reviews play a pivotal role in shaping product perception and influencing purchasing decisions. With the proliferation of online bookstores, Amazon has amassed an immense repository of book reviews spanning nearly two decades. These reviews contain valuable insights, sentiments, and trends that can unlock a treasure trove of information for authors, publishers, and book enthusiasts. This project embarks on a journey to harness the power of data, employing a comprehensive workflow to dissect and understand the vast collection of Amazon Books Reviews. Our mission is to develop a scalable solution that allow us to discover patterns, sentiment trends, and hidden correlations within the world of book reviews. We leverage cutting-edge tools and technologies, including Hadoop, Spark, MongoDB, and Python libraries such as Pandas and Scikit-learn. In this report, we embark on a detailed exploration of our project, delving into each stage of our workflow, from initial data discovery and preparation to feature extraction, model building, and rigorous evaluation.

2 Workplan

3 Discovery

Team

Tools

Framing

4 Data Preparation

Data Collection

Hypothesis Generation

Data Cleaning

Data aggregation

The MapReduce job was created to perform the inner join operation on the "Data table" and the "Rating table" based on the title. The output of the MapReduce job is a single file containing the joined records from both tables.

Mapper

The Mapper script processes the input data line by line, where each line represents a distinct record. It transforms

these lines into a key-value structure, where the key corresponds to the book title, and the value contains the remaining content of the line.
Given that the Mapper deals with data from two distinct sources, it becomes crucial to distinguish between records belonging to the 'Data table' and those in the 'Rating table'. This distinction is essential because it mandates a specific order of processing records from the 'Data' table need to be joined with corresponding records from the 'Rating' table in the Reducer phase. Consequently, the Reducer should process 'Data' table records before 'Rating' table records. To ensure this orderly processing, the Mapper augments the key with a special character for each table type. Specifically, it appends a hyphen ('-') as the second key element for records from the 'Data table' and 'www' for records from the 'Rating table.' By doing so, and thanks to Hadoop's sorting task made after, the Mapper guarantees that 'Data table' records are encountered and processed prior to 'Rating table' records during the subsequent phases of MapReduce.

Reducer

The Reducer script is responsible for processing the intermediate output records generated by the Mapper. Its primary role is to perform the join operation between the 'Data' table and the 'Rating' table, taking advantage of the pre-sorting of records by title. During its execution, the Reducer reads the records in a sequential order. As it encounters a record from the 'Data' table, it stores the information in one variable. Conversely, when it comes across a record from the 'Rating' table, it stores that information in another variable. Once both 'Data' and 'Rating' records for the same title are available, the Reducer performs the join operation by combining the data from these records.

MongoDB loading

5 Local Hypotheses Testing

Hypothesis 1

Reviews with longer text have higher helpfulness ratings.
Metric: Correlation coefficient (e.g., Pearson's correlation) between review length and helpfulness score.
Missing Values:

- *'review/text':* remove the entire sample
- *'review/helpfulness':* remove the entire sample

Data Transformation:

- *'review/text':* Count the number of words in each review removing punctuation and stopwords.
- *'review/helpfulness':* $helpfulness = \frac{x}{y}\sqrt(y)$

Description and Results The data cleaning and 'review/helpfulness' transformation process was performed using the *'pymongo'* library to exploit the MongoDB efficiency. Specifically, we defined a pipeline to perform the needed operations. As regards the 'review/text' transformation, we used the *'nltk'* library to tokenize the text, remove punctuation, stopwords and eventually count the number of words.
The correlation coefficient between the two variables is 0.3313 with a p-value < 0.05, indicating a statistically significant correlation.
A graphical confirmation is provided by Figure **??**. Indeed there is a positive correlation until around 400 words, after which the boxplot stabilizes. Thereby, we decided to analyze the correlation within the review length groups. As a results (Table 5) we got that the correlation is positive and statistically significant for the reviews with length between 0 and 400 words, while it becomes negative and statistically significant for the reviews with length greater than 750 words. As regards the reviews in the middle (i.e. between 400 and 750 words), the correlation is negligible.
Conclusion: The hypothesis is confirmed, but the correlation is not very strong and changes depending on the review length.

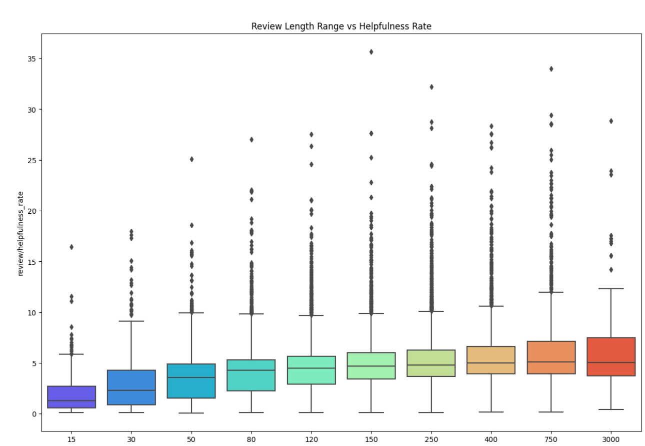

Figure 1. Correlation between review length and helpfulness score for different review length groups

Table 1. Correlation Coefficients and P-values for Different Groups

Group Number	Correlation Coefficient	P-value
400	0.2216	0.0000
750	-0.0188	0.2585
3000	-0.1418	0.0065

Hypothesis 2

Hypothesis 3

Hypothesis 4

Hypothesis 5

Hypothesis 6

6 Spark Hypotheses Testing

7 Helpfulness Prediction

8 Conclusions