Sharif University of Technology
Department of Mathematical Sciences

Regression and Data Analysis

User Engagement: A Data-Driven Analysis of Digikala

Authors:

Nima Azar
Aref Namayandeh
Hananeh Moballeghtohid

8th June, 2024

Project No. 02

Acknowledgement

First and foremost, We would like to express my sincere gratitude to Dr. Kasra Alishahi and Alireza Kadivar, whose expertise, guidance, and support were invaluable throughout this project. Their insightful feedback and encouragement helped shape the direction and outcome of this study.

We would like to extend our heartfelt thanks to our friend Sina Ghasemi Nezhad for sharing his computational power (GPU) with us in the last moments, which greatly accelerated our data processing and analysis.

Furthermore, We are grateful to the "Rade AI" team for providing the datasets and the necessary resources to conduct this research. The data was essential in performing a comprehensive analysis of user engagement metrics.

Abstract

In the fast-paced world of e-commerce, understanding user engagement is pivotal for the success and sustainability of online platforms. This project focuses on analyzing user engagement on Digikala, an e-commerce platform, by leveraging extensive datasets containing product information and user reviews. We employed advanced data analysis techniques, including sentiment analysis using a transformer-based model fine-tuned for Persian, to categorize user feedback into positive and negative sentiments.

Various methodologies were explored to calculate user engagement scores, such as simple scoring, weighted scoring, and a nuanced biased scoring approach that considers individual comment characteristics and sentiments. Our analysis revealed significant insights into user preferences and behavior, providing a comprehensive understanding of engagement dynamics on the platform.

Additionally, the clustering analysis identified distinct groups of user behaviors, allowing for targeted strategies to enhance user satisfaction. It is noteworthy that the weighted score and simple score exhibited a linear relationship, making them unsuitable for clustering based on categories. Conversely, the distribution of the total biased score offered a more effective basis for clustering, highlighting nuanced differences in user engagement.

This project underscores the importance of selecting appropriate metrics and methodologies for user data analysis and demonstrates the efficiency gains achieved through the use of GPUs for computationally intensive tasks. The findings offer valuable recommendations for enhancing user experience and optimizing product offerings on e-commerce platforms. Detailed code and additional resources are available on our GitHub repository.

Contents

1 Introduction

In the fast-paced world of e-commerce, user engagement is a critical factor for the success and sustainability of online platforms. User engagement, which encompasses various interactions and behaviors exhibited by users, is a key indicator of customer satisfaction, loyalty, and overall platform performance.

Understanding the importance of user engagement, businesses strive to define and measure meaningful metrics that capture user interactions and sentiments. Metrics such as review length, the number of likes and dislikes, and the frequency of mentioned advantages and disadvantages provide valuable insights into user preferences and behaviors.

This study aims to analyze user engagement on the e-commerce platform Digikala. By leveraging comprehensive datasets that include product information, user reviews, and other relevant variables, we seek to understand the dynamics of user engagement within the platform.

A central aspect of our analysis is the introduction of a novel parameter representing user engagement for each product. This parameter, derived from key engagement metrics such as review length and likes, allows us to assess and compare the engagement levels across different products.

We will also analyze this user engagement parameter in relation to various platform attributes such as product category, price and seller. Through statistical analysis, we aim to uncover the relationships between user engagement and these attributes, identifying patterns and trends that can inform strategic decision-making and platform optimization.

Our goal is to provide valuable insights and recommendations to enhance user experiences, optimize product offerings, and foster deeper connections with customers. By understanding the interplay between user engagement metrics and platform attributes, we hope to empower e-commerce platforms to achieve sustainable growth and success in an increasingly competitive digital landscape.

2 Data

This section describes the datasets used in this study, along with the preprocessing steps, feature engineering techniques, and sampling methods employed. The data was collected, cleaned, and transformed to ensure it was suitable for analysis and to provide meaningful insights into user engagement on the e-commerce platform.

2.1 Data Collection

The datasets used in this study were collected by the "Rade AI" team and published on LinkedIn. These datasets comprise over 1.2 million products and 6 million user reviews. The product dataset includes attributes such as price, rating, and brand, while the review dataset includes comment text, user ratings, and recommendation status.

2.1.1 Comments Table

The comments table includes detailed information about user reviews on various products. The columns in the comments table are as follows:

- **id**: Unique identifier for each comment.
- **title**: The title of the comment.
- **body**: The full text of the comment. (required)
- **created_at**: The date the comment was created.
- **rate**: The rating given by users, on a scale of 1 to 5, 0 if the user didn't participate in the rating.
- **recommendation_status**: Indicates whether the user recommended the product. (values can be recommended, not_recommended, no_idea and nan)
- **is_buyer**: Boolean indicating if the commenter purchased the product.
- **product_id**: The unique identifier of the product being reviewed.
- **advantages**: A list of advantages mentioned by the user.
- **disadvantages**: A list of disadvantages mentioned by the user.
- **likes**: The number of likes the comment received.
- **dislikes**: The number of dislikes the comment received.
- **seller_title**: The title of the seller.
- **seller_code**: The code identifying the seller.
- **true_to_size_rate**: A rating indicating how true the product size is to the described size.

2.1.2 Products Table

The products table provides detailed information about the products available on the platform. The columns in the products table are as follows:

- **id**: Unique identifier for each product.
- **title_fa**: The title of the product in Persian.
- **Rate**: The average rating of the product.
- **Rate_cnt**: The count of ratings the product has received.
- **Category1**: The primary category of the product.
- **Category2**: The secondary category of the product.
- **Brand**: The brand of the product.
- **Price**: The price of the product.
- **Seller**: The name of the seller.
- **Is_Fake**: Boolean indicating if the product is identified as fake.
- **min_price_last_month**: The minimum price of the product in the last month.
- **sub_category**: The sub-category of the product.

2.2 Data Cleaning, Preprocessing and Feature Engineering

Data cleaning and preprocessing steps included:

- Replacing values in the `advantages` and `disadvantages` columns with their respective counts for further calculations related to user engagement.
- Handling missing values and "nan" strings in various columns by replacing "nan" with 0 where appropriate, for further calculations.
- Converting Persian dates (which were in an unusual format) in the `created_at` column to a standard format of Jalali dates using a custom function.
- Standardizing the `recommendation_status` column to numeric values, with "recommended" and "not_recommended" set to 1, "no_idea" set to 0.5, and all other values set to 0, for further calculation of users engagement.
- Duplicate entries in the products table were identified and removed (based on Rate_cnt) to ensure each product is unique.
- The text in the `body` column was replaced by the number of words in each review.
- Binary indicators for the presence of titles in the `title` column: Titles were transformed into binary values indicating their presence (1) or absence (0).

2.3 Sampling

A stratified sampling method was used to create a representative sample of the comments table. The sample size was determined by dividing the total number of rows by 20 and rounding up to the nearest whole number. This sample was taken without replacement, ensuring that each row could only be selected once. This approach ensures the sample reflects the original distribution of user ratings, allowing for generalizable insights.

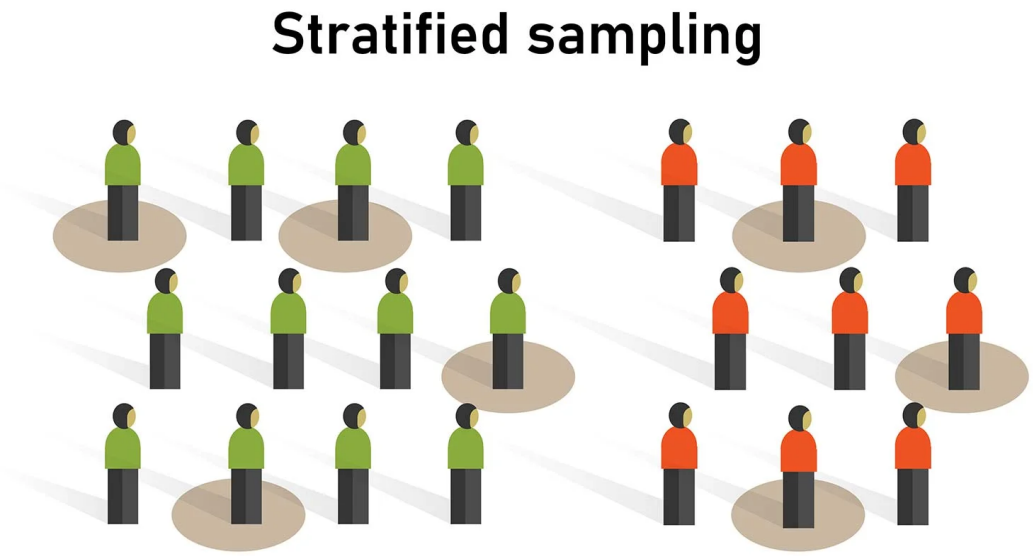


Figure 2.1: Stratified Random Sampling Example

This sampling method was primarily employed to facilitate sentiment analysis of the comments, which is computationally intensive. By working with a representative sample, we can obtain insights more quickly while reducing the computational resources required.

3 Sentiment Analysis

3.1 Introduction

In this chapter, we describe the sentiment analysis conducted on the user comments dataset. Sentiment analysis is a crucial aspect of understanding user engagement, as it helps in determining the overall sentiment expressed in user reviews. We employed a transformer-based model to perform sentiment analysis, leveraging its advanced capabilities for natural language understanding.

3.2 Overview of Transformer-Based Models

Transformer models have revolutionized the field of natural language processing (NLP). Introduced by Vaswani et al. (2017), these models use self-attention mechanisms to process text in parallel, making them highly efficient and effective for various NLP tasks. Among the most popular transformer models is BERT (Bidirectional Encoder Representations from Transformers), which leverages deep bidirectional representations to achieve state-of-the-art performance in numerous NLP tasks, including sentiment analysis.

3.3 Sentiment Analysis Methodology

3.3.1 Model Selection

For our sentiment analysis, we used the "HooshvareLab/bert-fa-base-uncased-sentiment-digikala" model. This model is specifically fine-tuned for sentiment analysis in Persian, making it suitable for analyzing comments from the Digikala e-commerce platform.

3.3.2 Batch Processing

The comments were processed in batches to efficiently utilize computational resources. The text data was tokenized and encoded into the appropriate format for the BERT model. Sentiment predictions were made for each batch of comments, outputting both the sentiment labels (positive or negative) and the confidence scores.

body	rate	label	confidence
خیلی خوبه	3	positive	0.94936776
توصیه میکنم	0	positive	0.9516735
اصلا چیزی که فکر می کردم نبود و بوی نسبتا بدی داشت قبلا عطر کرید گرفته بودم خیلی خوب بود ولی این کلا فرق داشت	3	negative	0.90547985
کاربردی و مناسب ولی جنس دسته ها شکننده	0	positive	0.576444
من مشکی خریدم بد نیس فقط فرمش یکم بهم ریختس که نسبت به قیمت خوبه.	3	positive	0.52019954
جالب و مناسب برای نوجوانان	3	positive	0.8996791
نسبت ب سایر عطراي جيبي كيفيت بسيار بالاتر دارد و در مورد رايحه هم خاصه تكراري نيس ويراي همه فصول مناسبه و براي افرادي ك نميتونن عطر بزرگ رو حمل كنن براي استفاده روزانه توصيه ميشه	0	positive	0.8691352
سلام .جوجه از نظر شکل خیلی مورد علاقه دخترم هست ولی حیف که یکی از پاهاش شکسته و وقتی میخواد جلو بره با نوک روی زمین میاد.سایزش هم خیلی کوچک بود	3	positive	0.7083973
خوبه ،نسبت به برندهاي ديگه ايراني	0	positive	0.7754733
هنوز استفاده نکردم قیمتش خوبه	3	positive	0.87917227

3.4 Sentiment Classification

The sentiment for each comment was determined based on the model's predictions. Comments with a rating of 1 or 2 were labeled as "negative", and those with a rating of 4 or 5 were labeled as "positive". For comments with ratings of 0 or 3, the sentiment was determined by the model.

3.5 Computational Power Required

Sentiment analysis, especially with large datasets, requires significant computational resources. Analyzing the entire dataset of 6 million comments with a powerful CPU would take approximately 170 hours. To expedite this process, GPUs (Graphics Processing Units) are often used for their ability to handle parallel computations more efficiently than CPUs.

In our study, we utilized a GPU to perform sentiment analysis on the comments. Using a GPU, it took approximately 6 hours to analyze 1.5 million comments. This significant reduction in processing time underscores the advantage of using GPUs for such computationally intensive tasks.

3.6 Summary

By using a transformer-based model fine-tuned for sentiment analysis in Persian, we were able to efficiently analyze the sentiments expressed in user comments on the Digikala platform. This analysis provided valuable insights into user sentiments, helping us better understand user engagement and preferences. Additionally, sentiment analysis allowed us to determine the user's sentiment for comments with a rate value of 0 (not participated in rating) or 3 (ambiguous sentiment), providing a more comprehensive understanding of user opinions.

4 User Engagement Score Calculation

4.1 Introduction

In this section, we explore various approaches to calculate the user engagement score for each product on the Digikala platform. The engagement score is a composite metric designed to capture the overall interaction and sentiment of users towards a product. We experimented with different methods to ensure that the score accurately reflects user engagement.

4.2 Comments Length Analysis

One of the key metrics considered for scoring was the length of the comments. The length of the comments, measured by the number of words, varied from 1 to approximately 980. The challenge was to categorize the comments into different groups and assign scores to each group. Here are the three methods we explored:

4.2.1 Heuristic-Based

We manually examined the distribution of comment lengths and selected a few numbers to categorize the comments. Each category i was assigned a score i.

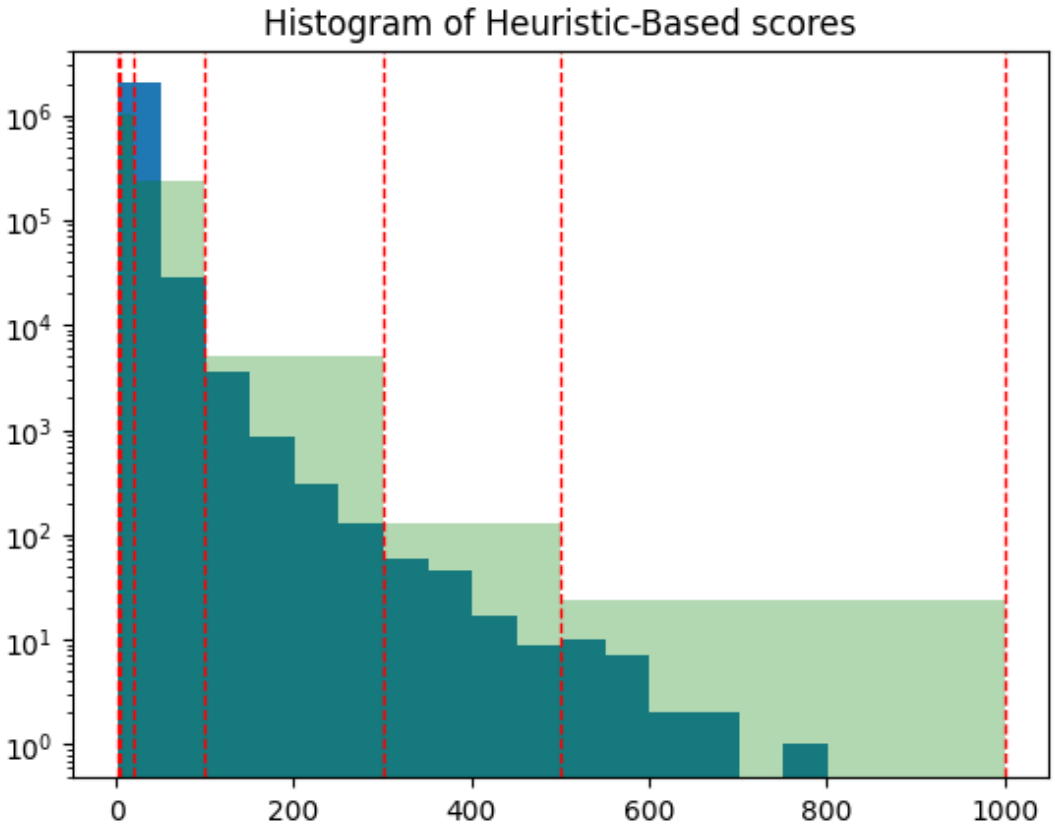


Figure 4.1: Heuristic intervals are 1, 5, 20, 100, 300, 500 and 1000 words

4.2.2 Percentile-Based

We calculated the 10th, 25th, 50th, 75th, and 90th percentiles of the comment lengths and used these values to define the categories.

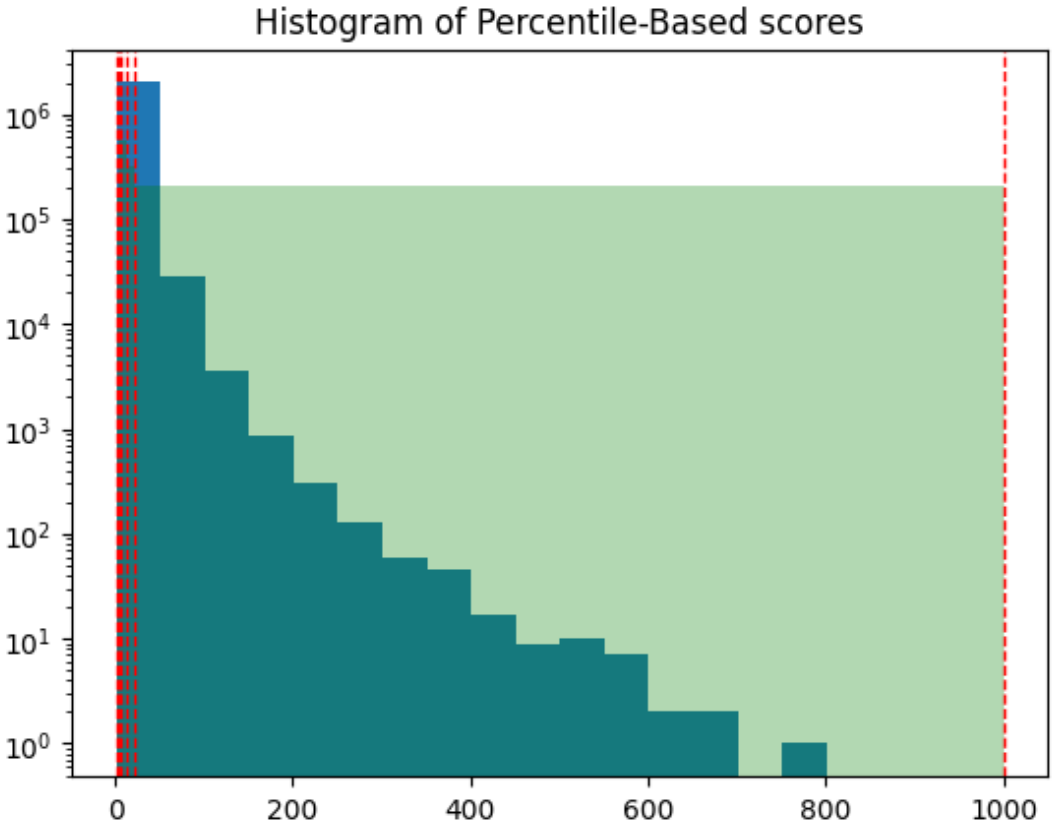


Figure 4.2: Percentiles are 10%, 25%, 50%, 75% and 90%

4.2.3 Clustering-Based (K-means)

We used the K-means clustering method to divide the data into several categories such that the total variance within the categories is minimized. This method helps in identifying natural groupings in the data.

For each method, we examined the categories, their counts, means, and variances. In the distribution plots, each red dashed line represents the category boundary, and the green histogram shows the frequency of comments in each category. Based on the analysis of the numbers and plots, we chose the third method (K-means) for categorizing the comments.

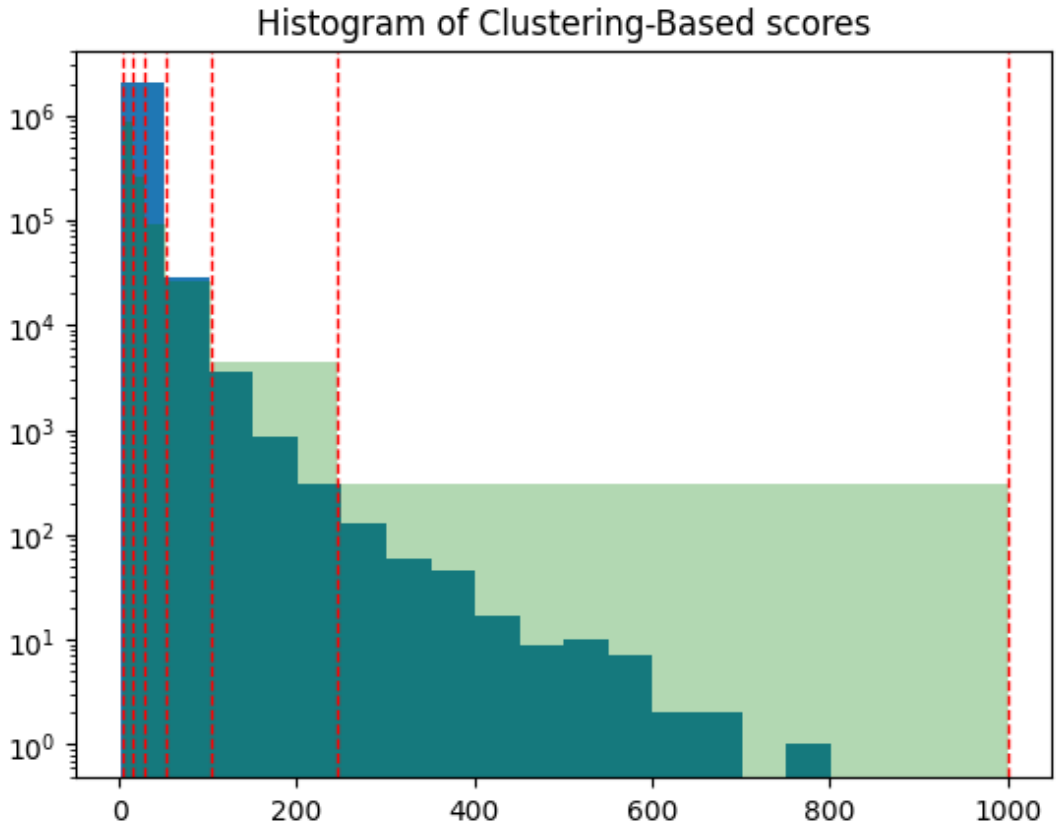


Figure 4.3: Intervals are 4.38, 14.02, 28.53, 52.72, 102.92, 244.65, 1000.

4.3 Scoring Approaches

We implemented several approaches to score the products based on the user comments and ratings:

4.3.1 Simple Scoring

This method assigns scores directly based on predefined criteria for comment length, ratings, and sentiment.

- Filter the comments to include only those where the user is a buyer.
- Calculate the comment score based on clustering intervals and the length of the comments.
- Compute the scores for different factors:
  - Comment score (normalized by 1000)
  - Number of ratings greater than 0
  - Sum of titles
  - Sum of likes and dislikes

  - Sum of recommendation status
  - Sum of advantages and disadvantages counts

- Sum these scores to get the total score:

```
total_score = (comments_score + rates_score + titles_score +
likes_score + recommended_score + advantages_score)
```

- Compute the average score by dividing the total score by the number of comments and multiplying by 100:

```
average_score = (total_score / len(group)) * 100
```

4.3.2 Weighted Scoring

This method assigns different weights to various factors such as comment length, ratings, and sentiment to compute a more nuanced engagement score.

- Filter the comments to include only those where the user is a buyer.
- Define weights for different factors:

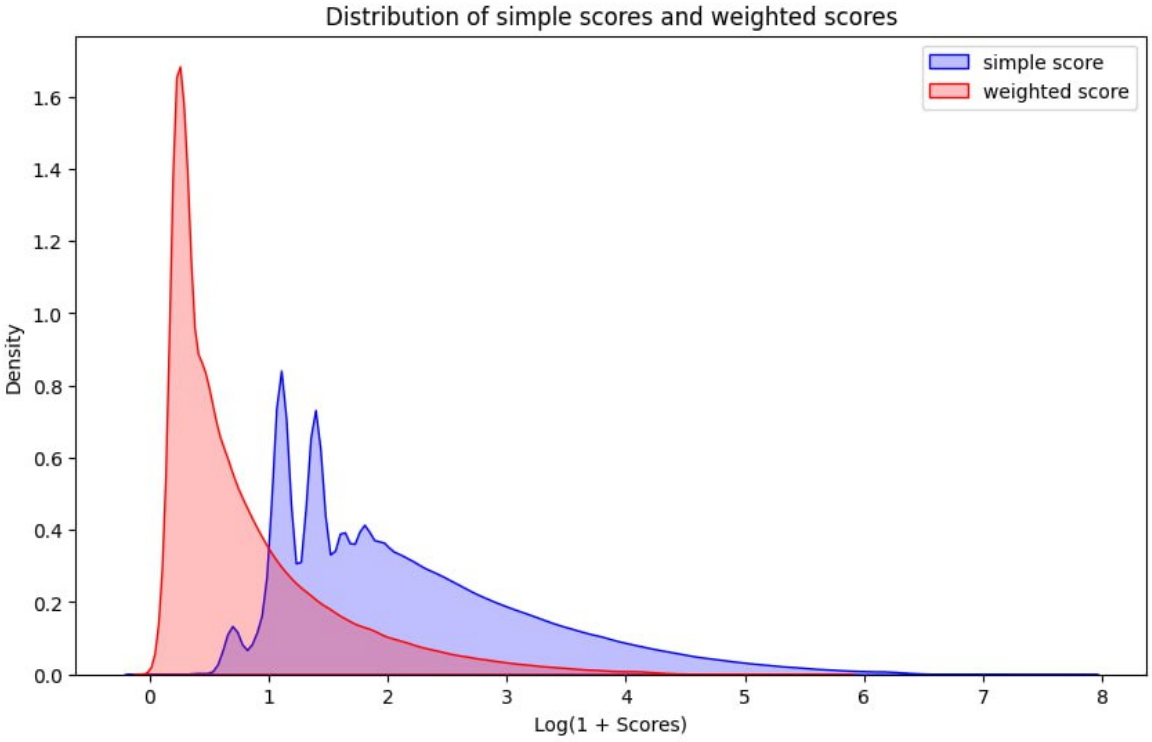
```
weights = {
"body": 10,
"title": 3,
"rate": 4,
"likes": 4,
"recommendation": 3,
"advantages": 5
}
```

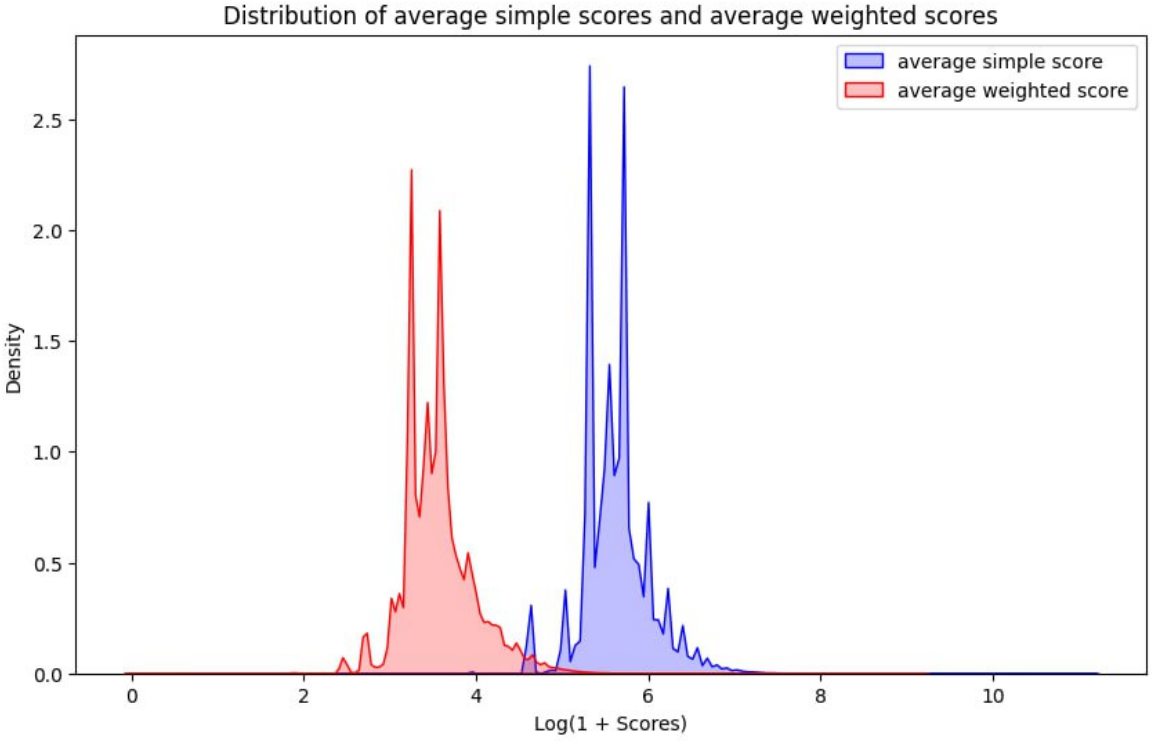
- Calculate the weighted scores for different factors:
  - Comment score (weighted by 10 and normalized by 100)
  - Number of ratings greater than 0 (weighted by 4)
  - Sum of titles (weighted by 3)
  - Sum of likes and dislikes (weighted by 4)
  - Sum of recommendation status (weighted by 3)
  - Sum of advantages and disadvantages counts (weighted by 5)
- Compute the total score by summing the weighted scores and normalizing by the sum of the weights:

```
total_score = (comments_score + rates_score + titles_score +
likes_score + recommended_score + advantages_score) / sum(weights.values())
```

- Compute the average weighted score by dividing the total score by the number of comments and multiplying by 100:

```
average_weighted_score = (total_score / len(group)) * 100
```


Distribution of simple scores and weighted scores
simple score
weighted score
Density
Log(1 + Scores)


Distribution of average simple scores and average weighted scores
average simple score
average weighted score
Density
Log(1 + Scores)

4.3.3 Biased Scoring

The `biased_score` function calculates a biased score for a group of comments, taking into account both positive and negative sentiments. The function aims to provide a more nuanced understanding of user engagement by considering individual comment scores and their sentiments.

Steps and Logic

- **Calculate Like/Dislike Ratio**:

```
likes = comment['likes'] + 1  # own comment
dislikes = comment['dislikes']
f = likes / (likes + dislikes)
```

  The function calculates the ratio of likes to the total number of likes and dislikes. An additional like is added to account for the user's own comment.

- **Compute Comment Score**:

```
score = body_score + rate_score + title_score + recommended_score + advantage_score
```

  The total score for the comment is computed by summing the individual scores.

- **Adjust for Sentiment**:

```
if comment['label'] == 'positive':
    pos_score = f * score
    neg_score = score - pos_score
else:
    neg_score = f * score
    pos_score = score - neg_score
```

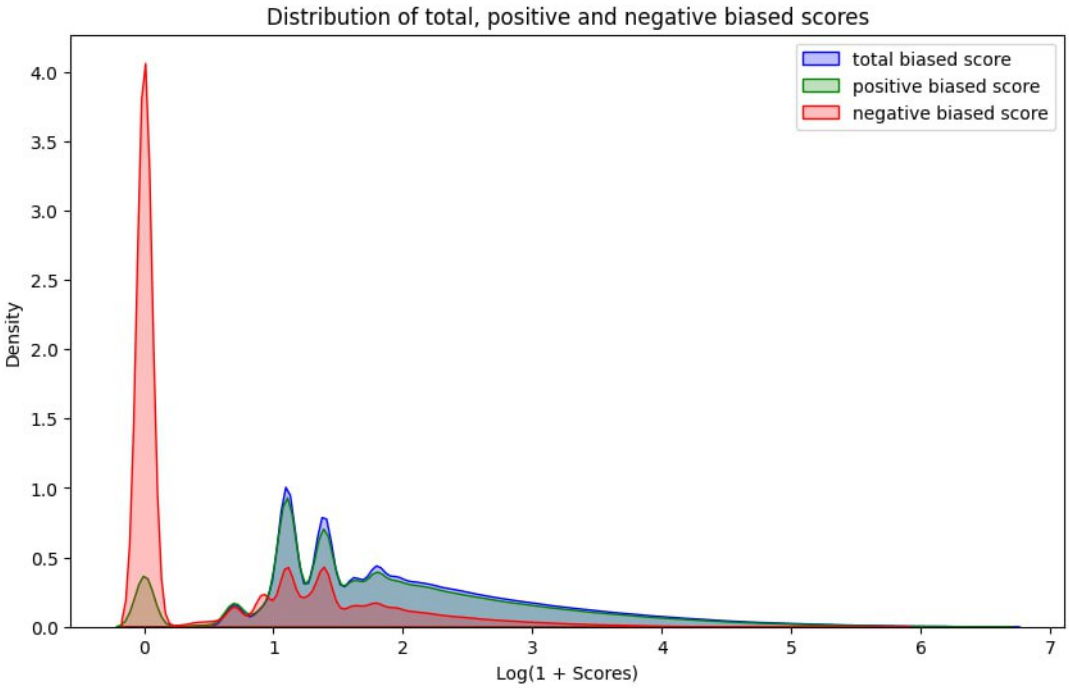
  Depending on the sentiment label of the comment, the score is split into positive and negative scores. The function uses the like/dislike ratio to weight the score appropriately.

- **Accumulate Scores**:

```
total_score += score
total_negative_score += neg_score
total_positive_score += pos_score
```

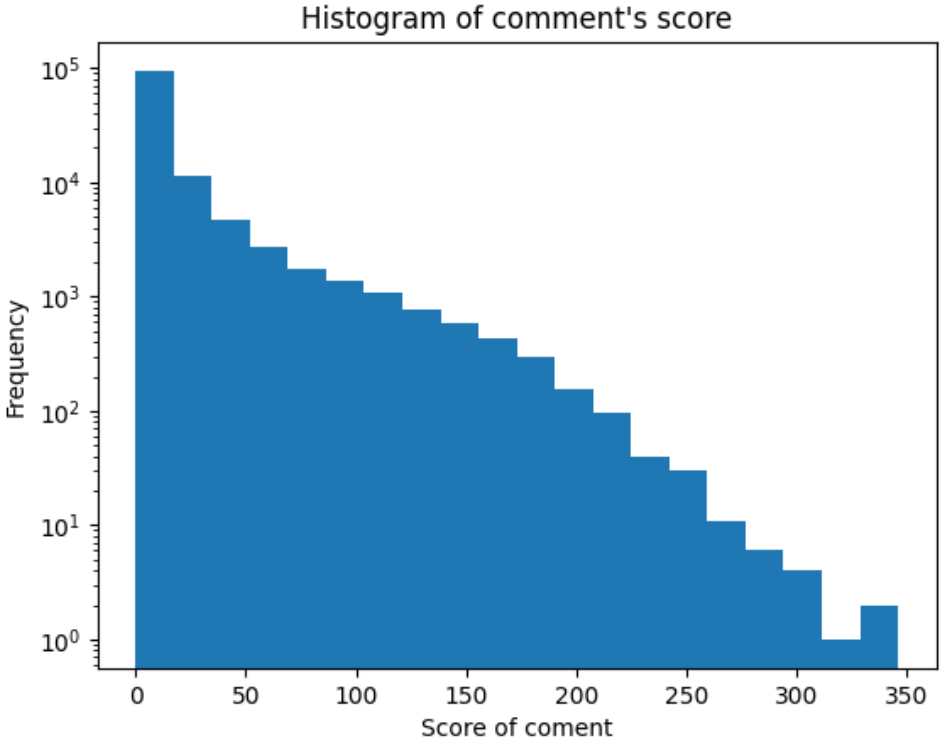
  The total score, total negative score, and total positive score are updated with the values calculated for each comment.

The `biased_score` function provides a comprehensive method for calculating user engagement scores by considering individual comment scores and their sentiments. It adjusts the scores based on the like/dislike ratio and separates them into positive and negative



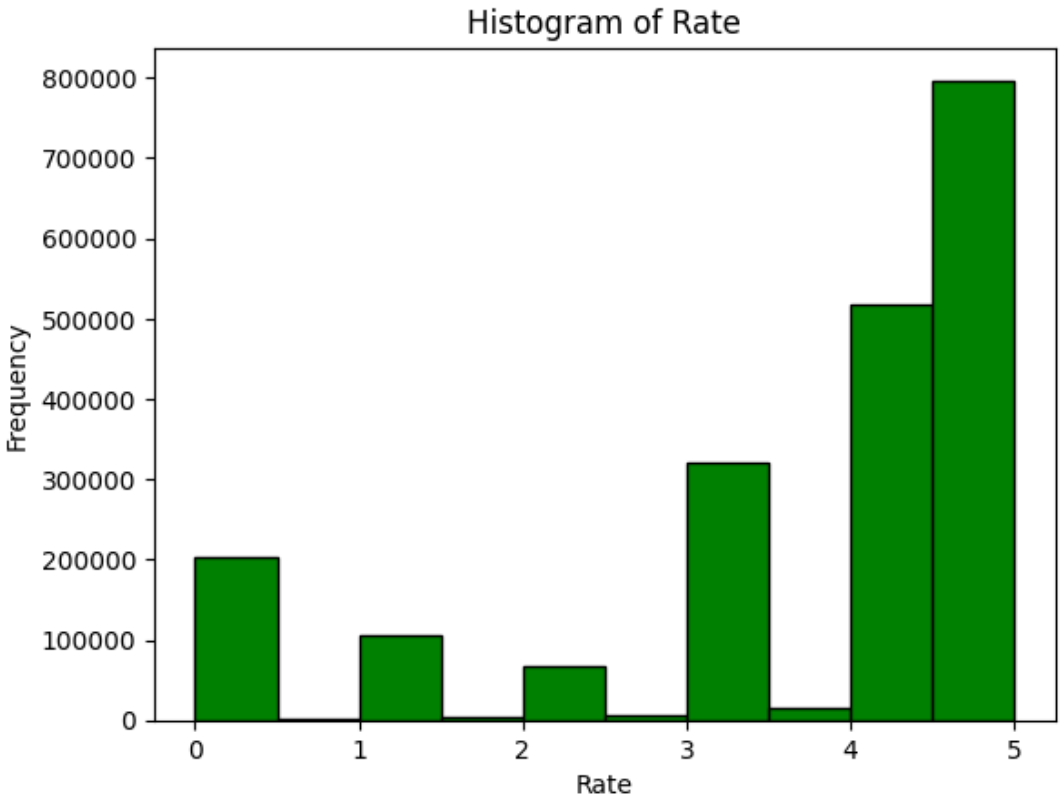
4.4 Distribution of Product Scores

After computing the scores using the different methods, we analyzed the distribution of product scores to ensure that they aligned with our expectations and provided meaningful differentiation between products.



4.5 Rates and Comments Grouping

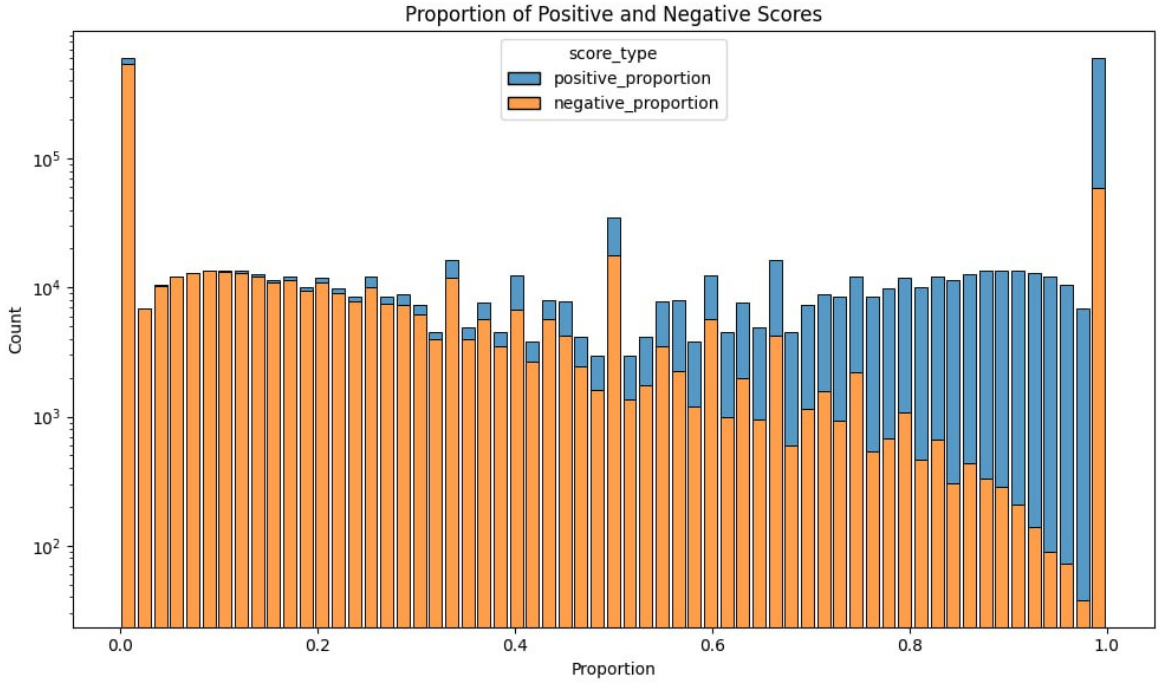
- **Rates Distribution**: We analyzed the distribution of ratings to understand the overall sentiment trends.



- **Labeling Comments**: Comments were labeled based on their ratings and sentiment analysis results, allowing us to categorize them more effectively.
  - Comments with a rating of 1 or 2 were labeled as ”negative”.
  - Comments with a rating of 4 or 5 were labeled as ”positive”.
  - Comments with a rating of 0 or 3 required further sentiment analysis to determine their sentiment.

4.6 Summary

The different approaches to calculating the user engagement score provided us with multiple perspectives on how to quantify user interactions with products.



This section highlights the importance of selecting appropriate metrics and methodologies in deriving meaningful insights from user data, and it underscores the value of exploring multiple approaches to validate the robustness of the calculated scores.

We suggest visiting GitHub/score_analysis for additional data tables and examples.

5 Clustering Analysis

5.1 Introduction

Clustering analysis is a key technique in unsupervised machine learning that groups a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups. In this section, we describe the clustering analysis performed on the user comments dataset to identify distinct groups of user behaviors and preferences.

5.2 Methodology

The clustering analysis involved several steps, including data standardization, determining the optimal number of clusters, applying the clustering algorithm, and interpreting the results. The following steps outline the process:

5.2.1 Data Standardization

Before applying clustering algorithms, it is essential to standardize the data to ensure that each feature contributes equally to the result. This involves scaling the features so that they have a mean of 0 and a

standard deviation of 1.

5.2.2 Determining the Optimal Number of Clusters

To find the optimal number of clusters, we used the Elbow method. This method involves running the clustering algorithm for a range of cluster numbers and plotting the within-cluster sum of squares (WCSS) against the number of clusters. The point where the plot starts to bend (elbow point) indicates the optimal number of clusters.

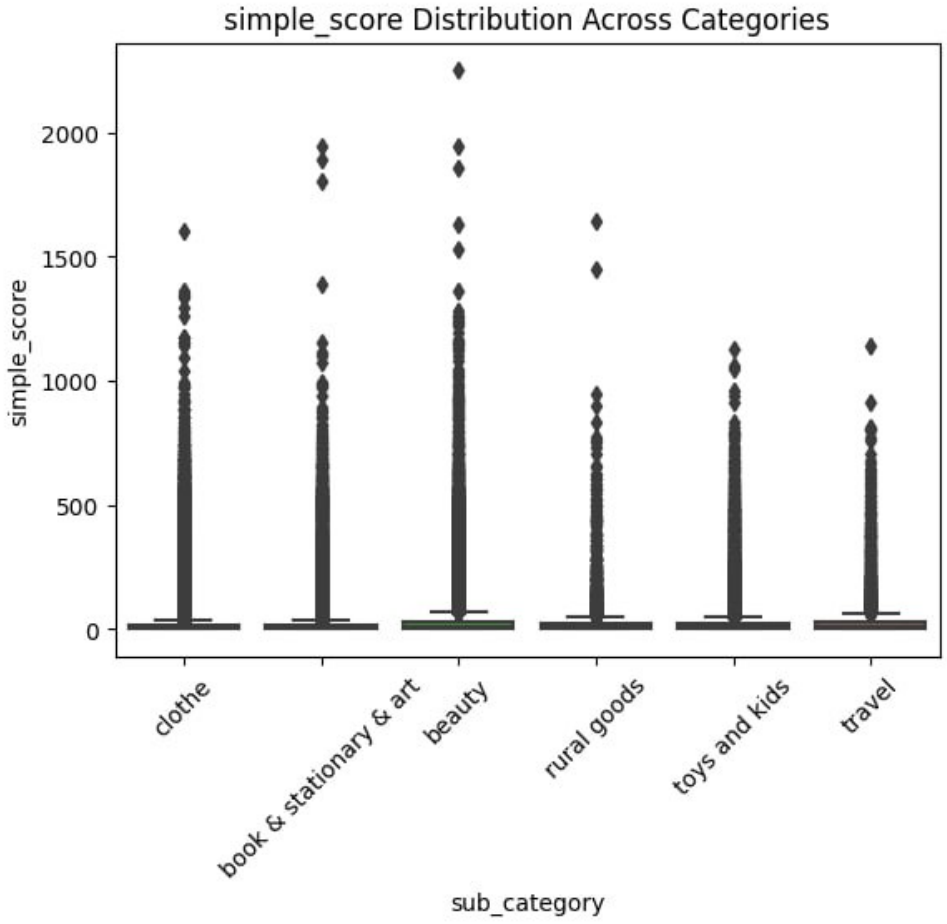
5.2.3 Applying the Clustering Algorithm

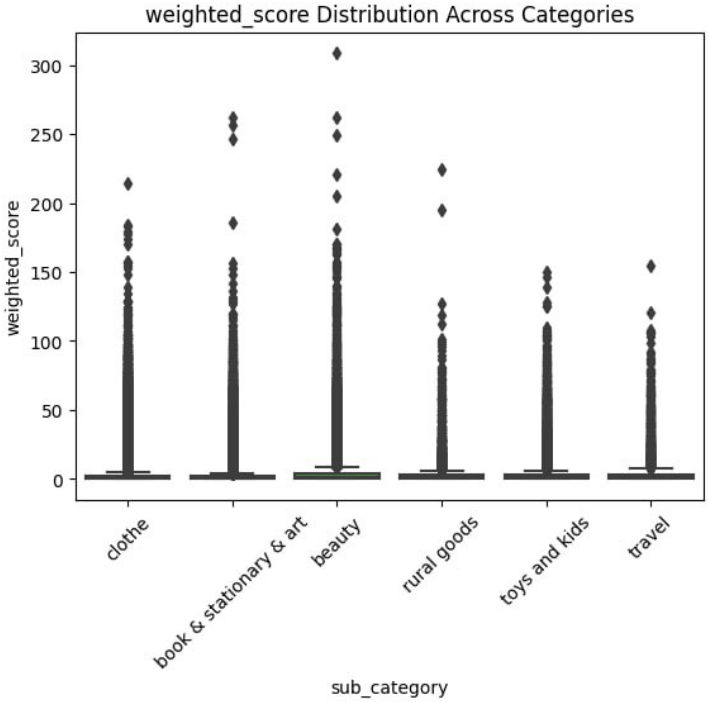
Based on the Elbow method, we chose the optimal number of clusters and applied the K-Means clustering algorithm to the standardized data. This resulted in each comment being assigned to one of the clusters.

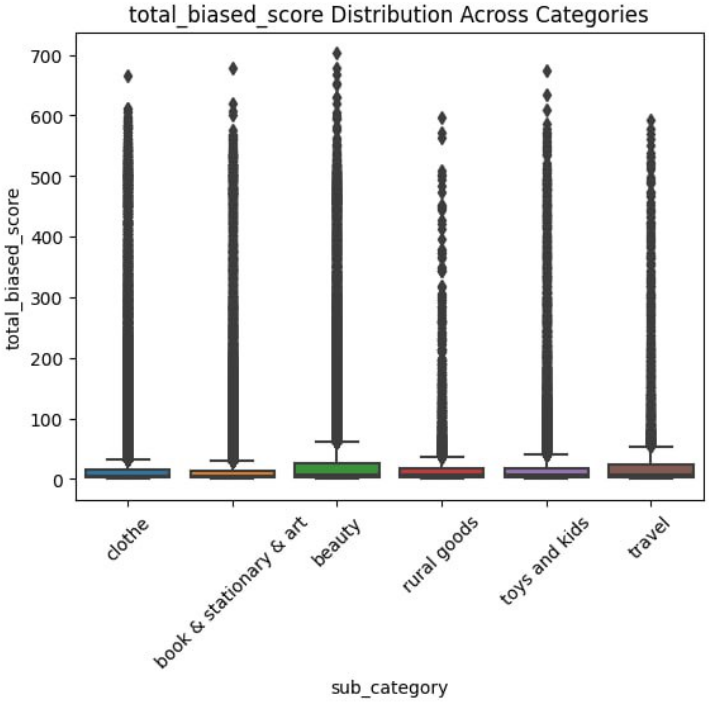
5.2.4 Interpreting the Results

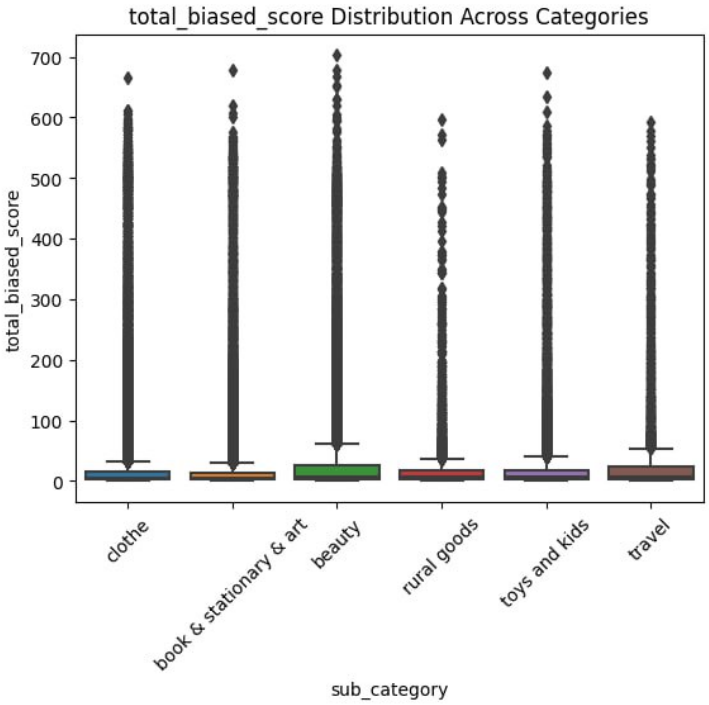
To interpret the clustering results, we examined the characteristics of each cluster by calculating the mean values of the features for each cluster. This helps in understanding the distinct properties of each group.

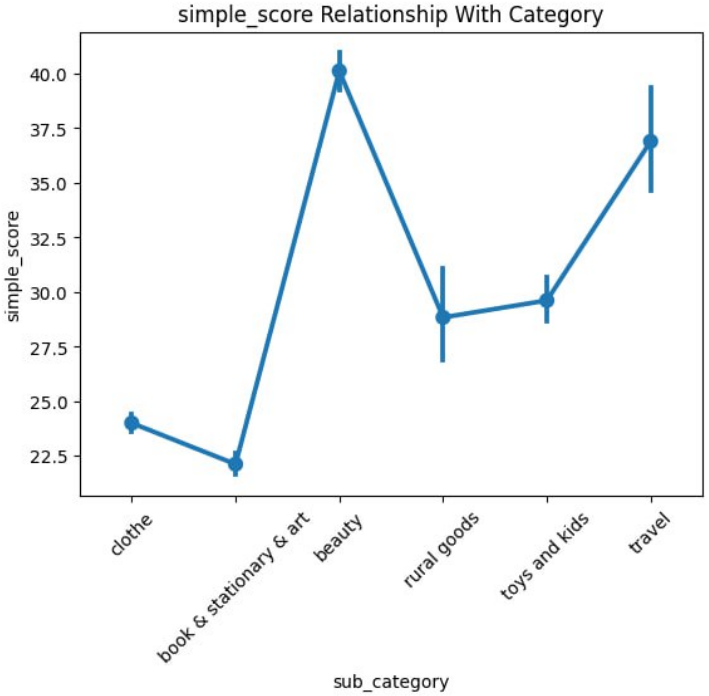
5.3 Results

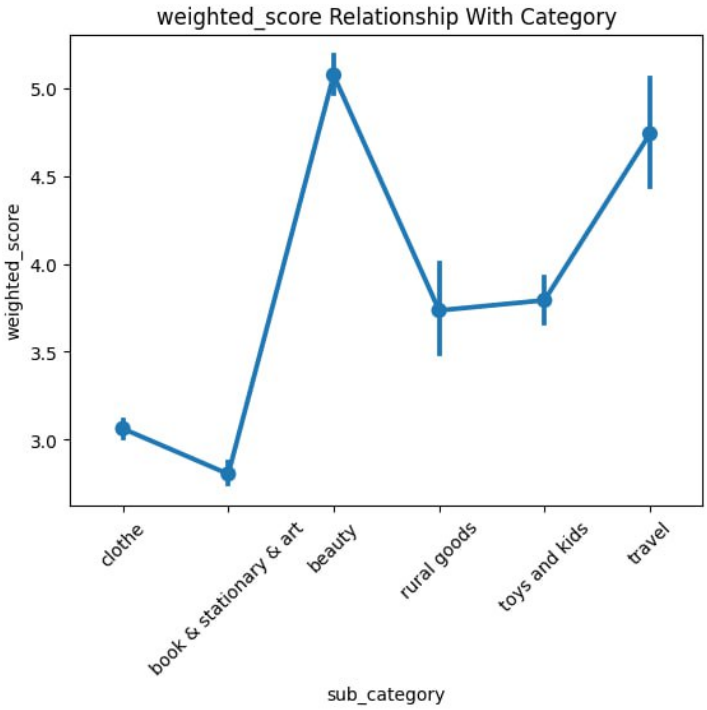


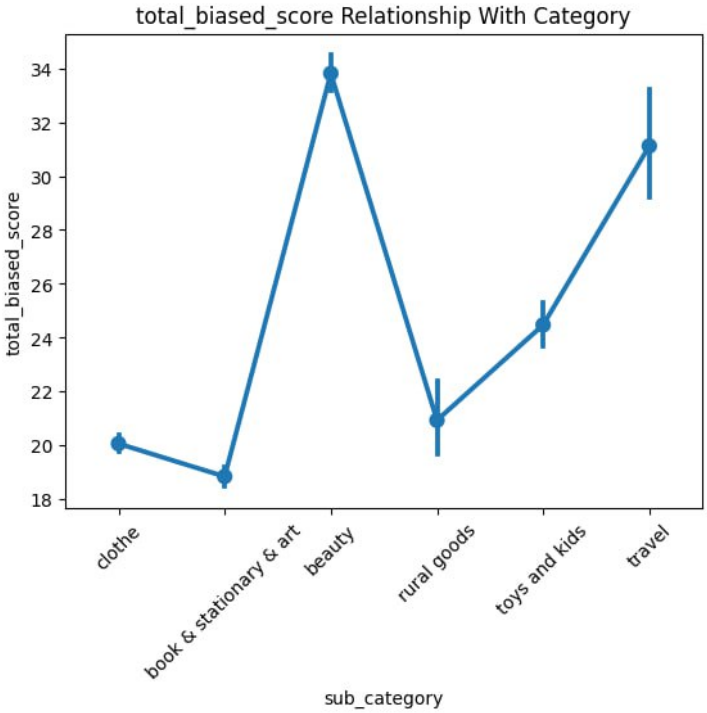

weighted_score Distribution Across Categories
weighted_score
300
250
200
150
100
50
0
clothe
book & stationary & art
beauty
rural goods
toys and kids
travel
sub_category

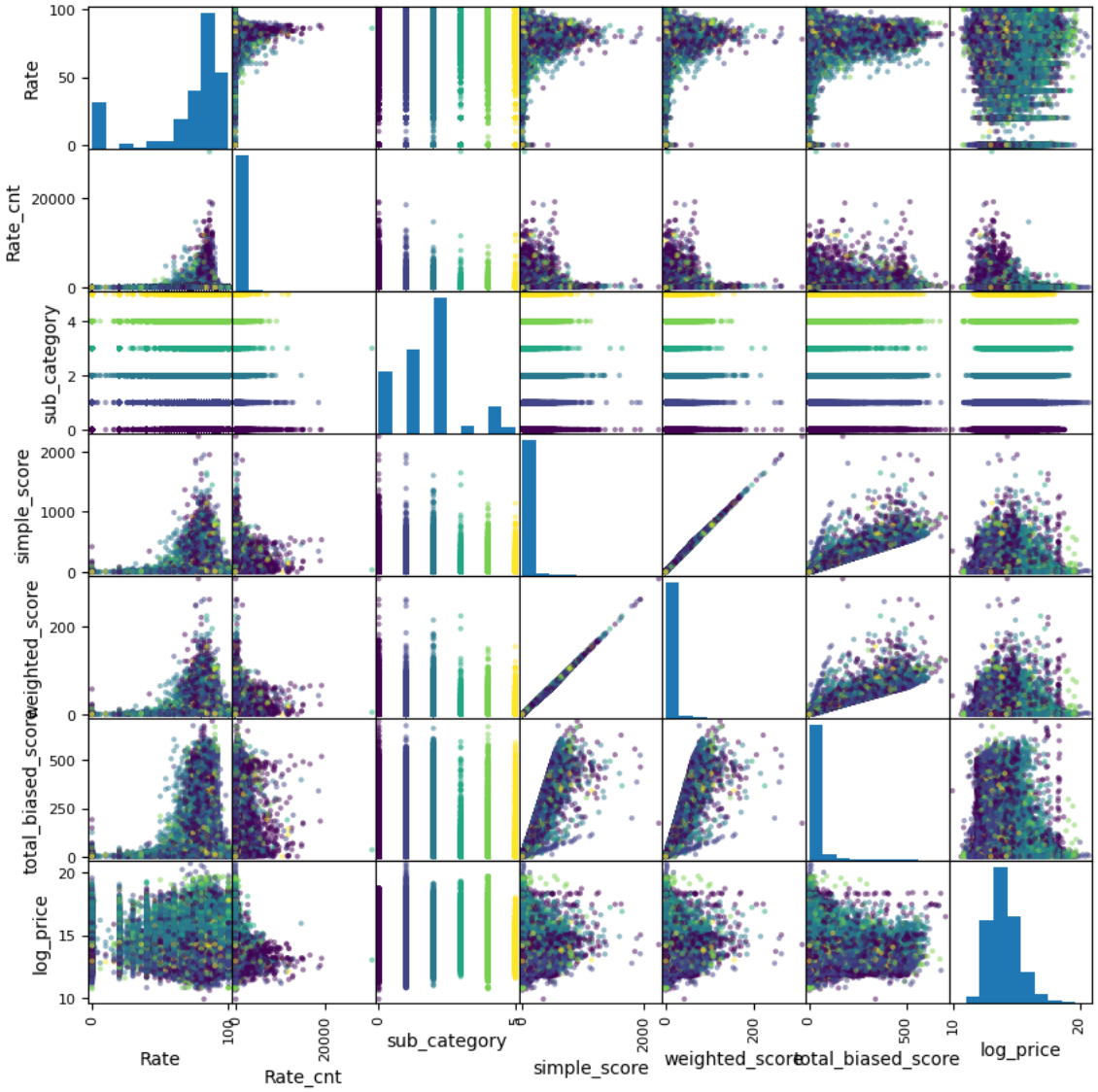

total_biased_score Distribution Across Categories
total_biased_score
700
600
500
400
300
200
100
0
clothe
book & stationary & art
beauty
rural goods
toys and kids
travel
sub_category


total_biased_score Distribution Across Categories
total_biased_score
700
600
500
400
300
200
100
0
clothe
book & stationary & art
beauty
rural goods
toys and kids
travel
sub_category


simple_score Relationship With Category
simple_score
40.0
37.5
35.0
32.5
30.0
27.5
25.0
22.5
clothe
book & stationary & art
beauty
rural goods
toys and kids
travel
sub_category


weighted_score Relationship With Category
weighted_score
5.0
4.5
4.0
3.5
3.0
clothe
book & stationary & art
beauty
rural goods
toys and kids
travel
sub_category


total_biased_score Relationship With Category
total_biased_score
34
32
30
28
26
24
22
20
18
clothe
book & stationary & art
beauty
rural goods
toys and kids
travel
sub_category



5.4 Conclusion

Clustering analysis provides valuable insights into the different types of user engagement on the platform. By understanding these clusters, we can tailor strategies to better meet the needs and preferences of different user groups. For instance, targeted marketing campaigns can be designed for highly engaged users, while strategies to increase engagement and satisfaction can be implemented for less engaged users.

It is important to note that the weighted score and simple score have a linear relation, and thus, clustering based on these scores did not reveal distinct categories. In contrast, the distribution of the total biased score provided a better basis for clustering, highlighting more nuanced differences in user engagement.

Ultimately, this analysis helps in enhancing the overall user experience on the Digikala platform by addressing the specific needs and behaviors of various user segments.