Paulina Thrasher
Data Engineering
Assignment 4

Executive Summary

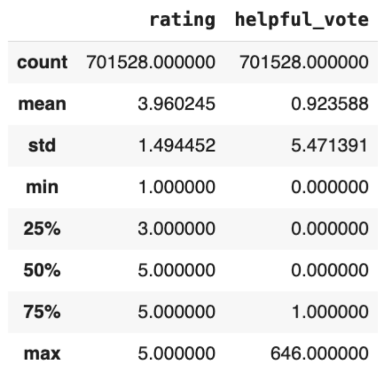
The following assignment utilizes the Amazon Reviews dataset created by McAuley et al. to show examples of basic exploration, preprocessing, and two examples of anomaly detection. Throughout the analysis, the category All_Beauty was used. This includes all products under the Beauty category in the dataset. Available columns include 'review_id', 'rating', 'title', 'text', 'images', 'asin', 'parent_asin', 'user_id', 'timestamp', 'helpful_vote', 'verified_purchase', 'review_date'. Section 1 gives an overview of the dataset, and some discussion of the anomaly detection methods used. Section 2 will provide a more detailed overview of the business insights and applications of these models.

Part 1: Data Exploration and Anomaly Detection

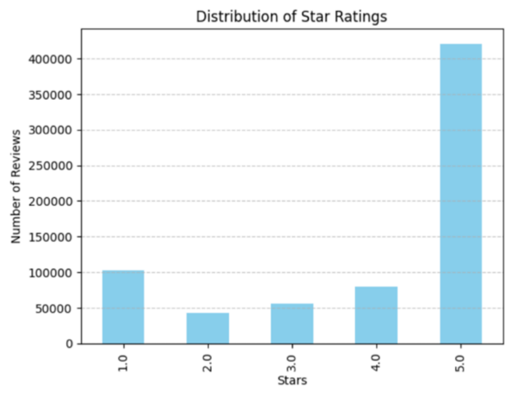
1.1: Data Exploration and Preprocessing

Basic Exploration

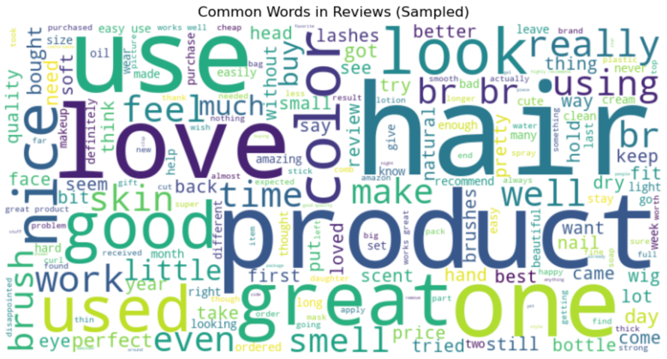
In total, there are 701,528 rows and 12 features. The descriptive statistics of the numeric columns 'rating' and 'helpful_vote' can be found below. The rating column gives information on the number of stars given to the product by the reviewer (1-5) and helpful_vote gives information on the number of votes that the review received by other users who deemed it helpful. The rating column shows a positive skew, with 75% of reviews rated 5 stars and a mean of 3.96. From these numbers, we also see that the number of helpful votes per review is highly skewed, with over half of the reviews receiving no votes, indicated by the median = 0 and mean = 0.92. The bar chart pictured shows a distribution of the star ratings in the dataset. The notable feature here is that most reviews fall into the 5-star category, confirming the earlier observation from the descriptive statistics.



	rating	helpful_vote
count	701528.000000	701528.000000
mean	3.960245	0.923588
std	1.494452	5.471391
min	1.000000	0.000000
25%	3.000000	0.000000
50%	5.000000	0.000000
75%	5.000000	1.000000
max	5.000000	646.000000



The review texts in the dataset vary significantly in length with the mean being approximately 33 words per review. However, a standard deviation of 46 suggests a high degree of variability. 75% of reviews are relatively short with 75% containing fewer than 40 words, the longest review contains over 2500 words. A frequency analysis of the most common words depicted in the word cloud below showed that reviews frequently include product related terms and expressions of sentiment. Notably, the word 'hair' appeared the most often with over 1300 instances found. This suggests that a significant number of reviews are related to haircare products which aligns with the category of focus for this assignment.



Preprocessing

Before further analysis, the dataset was preprocessed to ensure data quality and prepare for feature extraction First, missing review texts were dropped and missing values in the helpful_vote column were filled in with 0 to maintain consistency. Text data was cleaned by converting it to lowercase, removing non-letter characters, tokenizing, and eliminating English stop words using NLTK. Tokenization is the process of splitting text into smaller pieces called tokens. This is done to convert unstructured text into a list of elements that we can then count analyze or turn into vectors (Awan, 2024). NLTK is a Python library that is used commonly for natural language processing. It provides a library of stop words, which are commonly occurring words like "the", "is", and "and" that do not contribute meaningful information to the analysis (NLTK Project, 2024).

To address the skewness observed in the numeric features, log transformation was applied using log1p to compress the range and reduce the impact of outliers. This step ensured that these features were more normally distributed, which improves their suitability for downstream analysis and distance-based methods.

The cleaned reviews were then transformed into numerical feature vectors using TF-IDF. TF-IDF measures how often a word appears in a document relative to a collection of documents. The idea is that the importance is measured with respect to its frequency in a document and its rarity across multiple documents (GeeksforGeeks, 2025). In our case, a high value for a word means that it is used frequently in that review and its unusual in other reviews. TF-IDF vectorization applies L2 normalization to produce consistent input vectors for clustering and anomaly detection (Pedregosa et al., 2011). This means that after each review becomes a vector, each vector is scaled to the same length to prevent longer documents from dominating clustering.

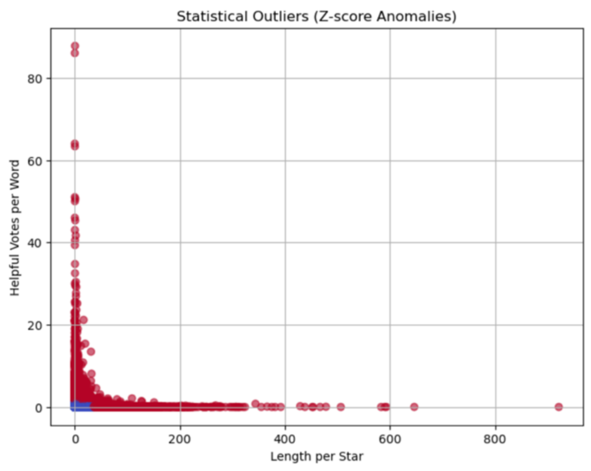
1.2 Anomaly Detection Implementation

Statistical Method

The first anomaly detection approach was a statistical method using Z-Scores. For this, several numerical features were engineered to capture atypical patterns in review behavior. These included the ratio of review length to rating (length_per_star), helpful votes per word (helpful_per_word), and helpful votes per star (helpful_per_star). Each of these were created with the goal to highlight reviews that might be disproportionally short, long, popular, or ignored by voters.

Z-scores were computed for each feature to measure how far a review deviates from the mean. Instead of evaluating features separately, a multivariate Z-distance was calculated using the Euclidean norm, and reviews exceeding the 99th percentile were flagged as anomalies. This captured reviews with unusual combinations of attributes, such as overly brief yet highly helpful reviews or lengthy ones with low engagement.

A scatter plot was used to visualize where each review fell using two of the computed features (pictured below, the plot shows Length per Star vs. Helpful votes per word). As expected, most points formed a cluster near the origin, while anomalous points, shown in red, appear further from the origin. Using the threshold of ±3 across these features resulted in approximately 2.74% of reviews flagged as statistical anomalies while the remaining 97.26% were classified as normal. To better understand which feature most often triggered the anomaly flag, each anomalous review was also labeled with the feature that had the highest absolute Z-score.

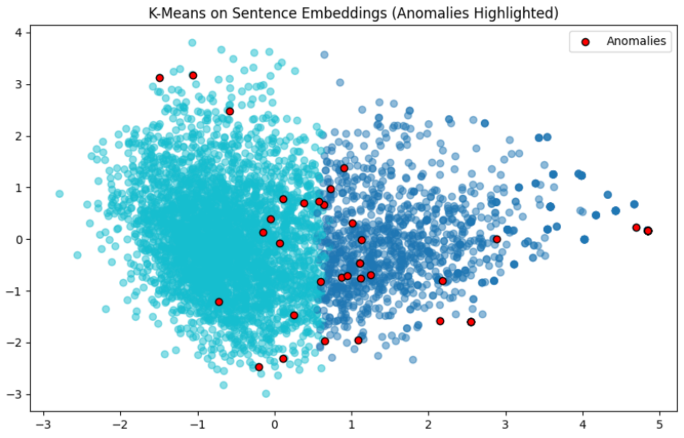


Clustering Based Method

The second anomaly detection method implemented was a clustering-based approaching using K-Means. The first model used the all-MiniLM-L6-v2 sentence transformer which captures semantic meaning beyond word frequency. To reduce the 384-dimensional embeddings and suppress noise, I applied Principal Component Analysis (PCA) to reduce the vectors to 50 components before clustering (Kaloyanova, 2024).
To determine the optimal number of clusters, I tested k values from 2 to 9 and selected the highest silhouette score. For this initial model, k=2 resulted in a score of 0.0663, suggesting weak but usable clustering separation. This is somewhat expected when it comes to short text data like product reviews, which often use similar vocabulary. After assigning cluster labels, I calculated the Euclidean distance of each review from its closest cluster center. Any review in the top 1% of distances were flagged as anomalies, as they were semantically distant from clusters.

To improve performance, I replaced the initial embedding model with paraphrase-MiniLM-L12-v2, which is optimized for capturing sentence-level semantic similarity (Reimers and Gurevych, 2019). This improved the silhouette score to 0.1297, indicating better defined clusters and stronger semantic separation between reviews. While still modest, this improvement confirms that higher-quality embeddings can improve anomaly detection in short text data.

The anomalies in the PCA scatter plot below are not entirely separate from the main data. This suggests that outlier reviews are not significantly different but are a bit unusual compared to other reviews. For instance, they may be shorter, oddly worded, or structured differently.



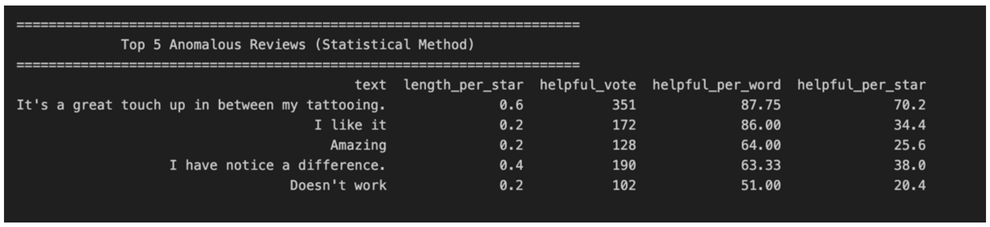
Part II: Evaluation and Business Insights

2.1 Evaluation

Comparison of Methods

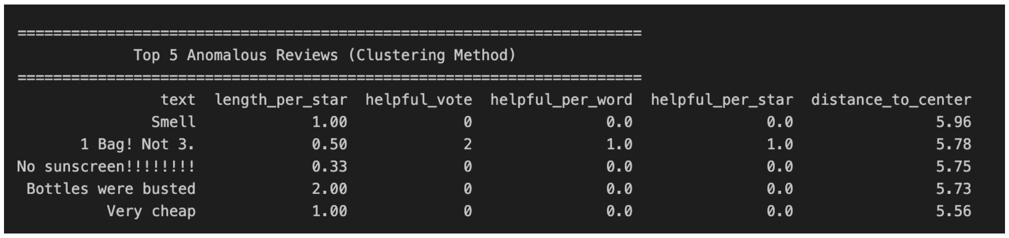
As described above, this project used two methods for anomaly detection in the Amazon beauty product reviews dataset. The first method was a statistical method using z-scores and the second was a clustering-based method using K-Means on sentence embeddings.

The statistical method flagged reviews with extreme numerical patterns such as unusually high helpful vote ratios in short reviews, a sample of 5 anomaly reviews can be seen below. For example, the review “I like it” received 172 helpful votes despite only containing three words, resulting in a highly unique helpful-per-word score. These types of anomalies suggested potential manipulation, such as inflated helpfulness scores.



```
======================================================================
             Top 5 Anomalous Reviews (Statistical Method)
======================================================================
                                     text  length_per_star  helpful_vote  helpful_per_word  helpful_per_star
It's a great touch up in between my tattooing.              0.6           351             87.75              70.2
                                I like it              0.2           172             86.00              34.4
                                  Amazing              0.2           128             64.00              25.6
                  I have notice a difference.              0.4           190             63.33              38.0
                               Doesn't work              0.2           102             51.00              20.4
```

In the second method, the clustering model flagged reviews that were semantically distant from other reviews using PCA-reduced sentence embeddings. This method starts by converting each review into an embedding, then simplifying these embeddings for faster analysis. These simplified embeddings then allowed the K-Means algorithm to group similar reviews, any reviews that were far from the centers of these clusters were flagged as anomalies. For instance, reviews like "Smell" or "Very cheap" were flagged because they were vague, short, or used different language from the other reviews.

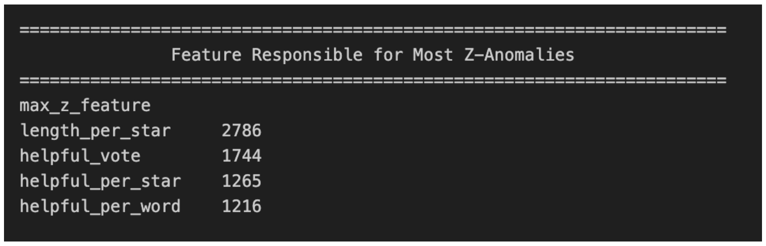


```
====================================================================
            Top 5 Anomalous Reviews (Clustering Method)
====================================================================
                text  length_per_star  helpful_vote  helpful_per_word  helpful_per_star  distance_to_center
               Smell             1.00             0               0.0               0.0                5.96
       1 Bag! Not 3.             0.50             2               1.0               1.0                5.78
No sunscreen!!!!!!!!             0.33             0               0.0               0.0                5.75
 Bottles were busted             2.00             0               0.0               0.0                5.73
          Very cheap             1.00             0               0.0               0.0                5.56
```

The two methods detect different types of anomalies with the statistical method identifying reviews with unusual quantitative characteristics and the clustering method flagging reviews that are linguistically unique. In terms of effectiveness, the statistical method is more transparent and interpretable, while the clustering method is better at detecting semantic qualities. The statistical method may miss contextually weird reviews, like vague or misleading ones that don't show extreme numbers but still feel off when you read them. On the other hand, the clustering method is limited by sentence embedding quality and dimensionality reduction. Sentence embeddings are used to capture meaning, but when dimensions are reduced using PCA, the nuance is diminished. Also, since clustering is unsupervised, it relies on distance from a cluster center to flag anomalous reviews, which may not always match human judgement.

Sensitivity Analysis

Using the multivariate Z-distance method, reviews were flagged as anomalies if their combined deviation across all engineered features exceeded the 99th percentile. This threshold resulted in approximately 1.00% of the dataset being flagged as anomalous. Among these, the most common contributing factor was length_per_star, indicating that unusually short or long reviews relative to their rating were frequently outliers (See image below for the columns with the highest z-scores). This approach was more conservative than the single-feature threshold method and reduced the risk of flagging cases that were borderline, highlighting a useful tradeoff between precision and coverage.



```
======================================================================
               Feature Responsible for Most Z-Anomalies
======================================================================
max_z_feature
length_per_star     2786
helpful_vote        1744
helpful_per_star    1265
helpful_per_word    1216
```

For the clustering method, PCA dimensionality was tested with 50 and 100 components. When using the all-MiniLM-L6-v2 model, increasing the components from 50 to 100 reduced the silhouette score from 0.0663 to 0.0530. This suggests that too many dimensions can introduce noise rather than improve cluster clarity. Similarly, when using the paraphrase-MiniLM-L12-v2 model, reducing the dimensionality to 50 components provided the best results, a silhouette score of 0.1297 compared to 0.1121 at 100 components. As mentioned above, this score indicates that clusters exist but are not very distinct, and this should be interpreted as weak separation. However, these tests show that more data complexity does not always improve results and that fine-tuning dimensionality is important when working with embeddings. Choosing the optimal number of components can help find a balance between preserving meaningful variation and reducing noise.

2.2 Business Insights

The two anomaly detection methods surfaced distinct types of unusual reviews that could present problems for an e-commerce platform. The statistical method primarily flagged short reviews with disproportionately high helpful vote counts. For instance, one review simply stated "I have noticed a difference" but had 190 helpful votes. These reviews stood out not because of what they said, but because the numerical influence they had did not match the content. Additionally, the most common trigger for statistical anomalies was length_per_star, once again pointing to overly short reviews with high ratings. This insight could help platforms adjust review weighting to reduce the influence of vague but popular content. The clustering method flagged reviews that were linguistically different from the rest of the dataset. Examples include vague or context-free comments like "Smell"," 1 Bag! Not 3. ", or "No sunscreen!!!!!!!!". These reviews were often short or lacked further detail about the product itself. Although they might seem harmless, their unusual language and structure made them outliers in the embedding space.

Several patterns emerged overall for anomalous reviews. These included extremely short reviews, high vote counts for low-effort content, emotionally extreme language, and minimal explanatory value. These reviews can be problematic for e-commerce companies in several ways. First, inflated helpfulness scores distort product rankings and mislead customers by signaling that a product is high-quality when it is not. Second, vague or reviews that lack context reduce the usefulness of the reviews overall, which can frustrate users and reduce trust in the platform. Finally, the algorithms that rely on review data to make recommendations or detect fraud may be thrown off by these anomalies, affecting both customer experience and backend systems. For instance, inflated helpful votes can cause the algorithm to overestimate a product's popularity. Also, anomalous reviews can cause bias in collaborative filtering, a common recommendation technique that looks at patterns between users and products (Murel & Kavlakoglu, 2024. If fake or low-quality reviews skew the ratings or descriptions of products, the algorithm may recommend these items to the wrong audience.

Citations

1. Awan, A. A. (2024, November 22). *What is tokenization?* DataCamp. https://www.datacamp.com/blog/what-is-tokenization
2. GeeksforGeeks. (2025, February 7). *Understanding TF-IDF (Term Frequency-Inverse Document Frequency)*. https://www.geeksforgeeks.org/understanding-tf-idf-term-frequency-inverse-document-frequency/
3. Kaloyanova, E. (2024, April 15). *How to combine PCA and K-means clustering in Python?* 365 Data Science. https://365datascience.com/tutorials/python-tutorials/pca-k-means/365 Data Science+3365 Data Science+3365 Data Science+3
4. McAuley, J., Hypolite, D., & Li, H. (2023). *Amazon Reviews 2023*. McAuley Lab, UCSD. Hugging Face. https://huggingface.co/datasets/McAuley-Lab/Amazon-Reviews-2023
5. Murel, J., & Kavlakoglu, E. (2024, March 21). *What is collaborative filtering?* IBM. Retrieved from https://www.ibm.com/think/topics/collaborative-filtering
6. NLTK Project (2024, August 19). *NLTK stopwords*. Natural Language Toolkit. Retrieved May 2, 2025, from https://www.nltk.org/search.html?q=stopwords
7. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … & Duchesnay, É. (2011). *Scikit-learn: Machine learning in Python*. Journal of Machine Learning Research, 12, 2825–2830. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
8. Reimers, N., & Gurevych, I. (2019). *Sentence-BERT: Sentence embeddings using Siamese BERT-networks*. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. http://arxiv.org/abs/1908.10084