Università degli Studi di Milano – Bicocca

# TEXT MINING & SEARCH

EXAM PROJECT:
*SUPERVISED AND UNSUPERVISED LEARNING ON AMAZON FINE FOOD REVIEWS*

**DATA SCIENCE MASTER'S DEGREE**

**2023/2024**

*Students:*

*Sara Campolattano – 906453*

*Induni Sandapiumi Nawarathna Pitiyage – 906451*

## TABLE OF CONTENTS

## ABSTRACT

In the era of digital commerce, i.e., e-commerce, online reviews have emerged as a critical source of consumer feedback and product insight. As a matter of fact, reviews reflect the trustworthiness of platforms, giving them exceptional power on the market. A key example is Amazon: it is the world's largest and most valuable e-commerce globally, with a market capitalization exceeding $1.7 trillion, that with its review system has an ace up the sleeves in influencing purchase decisions. In this study, we aim to analyze Amazon reviews through advanced text preprocessing, binary classification, and clustering techniques. Our approach starts with advanced text preprocessing to clean and normalize Amazon reviews, tackling issues like noise and language variability. We then employ machine learning algorithms for classification to categorize reviews. Lastly, we apply unsupervised methods, i.e., clustering with the aim of uncovering patterns and trends in the feedback.

## INTRODUCTION

Since day one, reviews have always had a big impact on customers' decisions to go through with purchasing a product and have now become an essential factor when talking about a product's visibility on the platform. With Amazon evolution, reviews have emerged as a vital source of authentic user feedback, and a significant number of consumers rely on them before making a purchase. It is indeed very common for products with an enormous number of negative reviews to never see the light of day again after the very first wave of customer feedback, as the persistent poor ratings can significantly diminish their visibility and attractiveness to potential customers. On the other hand, positive reviews are known to benefit sellers by improving product rankings and increasing the likelihood of sales. We therefore aim to develop classification and clustering models to gain valuable insights and to possibly enhance the management of consumer feedback.

# 1. DATASET INSPECTION

The dataset employed for this project is the Amazon Fine Food Reviews dataset, which is available on Kaggle [1] and, as the name says, it consists of reviews written by consumers about fine foods. The dataset comprises of the following features:

- ProductID: product unique identifier.
- UserID: user unique identifier.
- ProfileName: user profile name.
- HelpfulnessNumerator: number of users who found the review helpful.
- HelpfulnessDenumerator: number of users who indicated whether they found the review helpful or not.
- Score: review rating (between 1 and 5).
- Time: review date in UNIX format.
- Summary: review summary.
- Text: text of the review.

A first inspection of the raw data allowed for the definition of the number of reviews contained, that is, 568.454, and the number of reviewed products, which amounts to 74.258, implying an average of approximately seven reviews per product. Another interesting aspect discovered is the timeframe in which those reviews were written, going from 1999 (five years after its foundation) to 2012.

This primary glance at the data allowed also to discover some critical issues within the raw data that necessitated preprocessing and cleaning before proceeding with our tasks. One significant issue was the presence of duplicated reviews in the dataframe. These duplicates were identified as reviews written by the same user with identical ratings, timestamps, and text, but associated with different product codes. This issue can arise for various reasons, including users updating their feedback, system errors, accidental resubmissions, or reviewing the same product under different circumstances. These duplicated entries were therefore eliminated, as they do not provide additional informative value to our purpose.

Additionally, some reviews were found to have the same user ID and product ID but were posted at different times. These could be the result of multiple purchases over time or updates to reviews made in a prior time, potentially after the seller addressed an issue. Also in this case, most reviews contained almost identical text with only a different score, suggesting they were updates rather than new reviews. To maintain consistency and avoid potential complications in the analysis, all duplicated entries were therefore removed, and only the most recent reviews for each instance were retained. After removing such duplicates, the dataset then comprised of 392.969 entries.

Along with duplicates, missing values were also inspected, and it was discovered that only the feature "Summary" contained exactly 3 missing entries. As said feature is not considered for the analysis that was carried out, it was decided not to address such issue.

As previously mentioned, the reviews cover a time span of 13 years. However, from the inspection carried out it is evident that the majority of the reviews were written after 2008, as it can be seen from *Figure 1*.

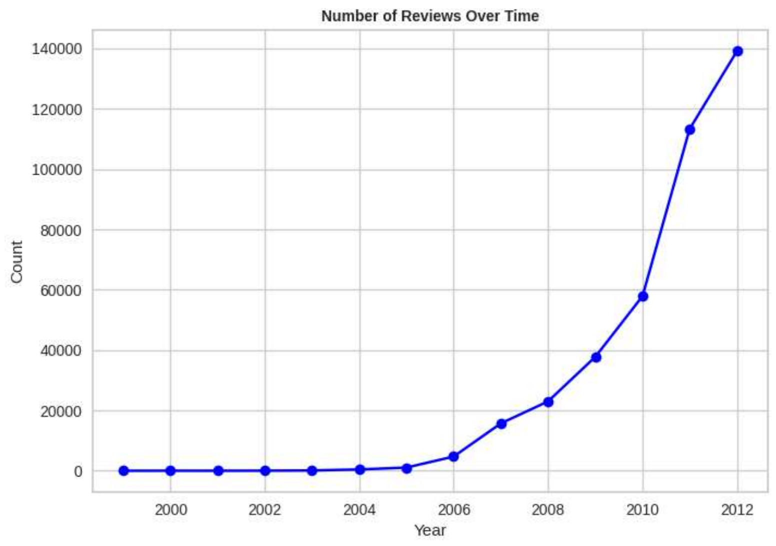


*Figure 1: Number of Reviews over time.*

It is therefore evident that an impressive growth characterized the reviews, which could be certainly due to the exponential popularity that Amazon gained over time and consequently its use.

Although this behavior was expected, it may imply that there could be a structural or semantic difference between reviews written back in the first years of the current millennium and those written over the end and after the first decade. This aspect will therefore be addressed later on in the Preprocessing section.

## 2. PREPROCESSING

As human-generated content, such as reviews, presents with various criticalities like noise, variability and complexity, preprocessing becomes an essential task to ensure the accuracy and effectiveness of the analysis that are going to be carried out. Therefore, to improve the quality and interpretability of the reviews for further classification and clustering tasks, in this section we will discuss the methods employed to preprocess the dataset, including normalization, stop-words removal, tokenization, stemming and the handling of special cases such as abbreviations.

### 2.1 NORMALIZATION

As previously mentioned, text preprocessing is crucial for clearing up ambiguities and guaranteeing the corpus' consistency. In this context, it was decided to perform, as first step, normalization since predefined stemming and stop-words removal tools that frequently use lowercase and might miss distinctive user-generated accents. Therefore, normalization consisted in:

- Lowercase Conversion**:** all text was converted to lowercase to maintain uniformity.
- Abbreviation Replacement: to preserve the critical word “not”, which is key for accurately interpreting binary reviews, common abbreviations were expanded using regular expressions (e.g., transforming “don’t” into “do not”, “isn’t” into “is not”).
- Accent Standardization: Accented characters were replaced with their standard counterparts to ensure consistency (e.g., converting “à” to “a”).

Additionally, reviews presented with special characteristics such as emojis, emoticons, hashtags, URLs and double white spaces. To address these issues, custom regular expressions (regex) were defined and applied for each specific case to effectively remove them.

### 2.2 STOP-WORDS REMOVAL

As stop-words are basically words that crop up repeatedly in texts but usually have no bearing on particular subjects or situations, it is clear that they don't add any significant insights and can therefore be eliminated to improve analysis. To address this, the default stop-words list provided NLTK package [2] was used, as it is a standard reference. However, we decided to make a few changes to the stop-words list to better suit our purposes. Specifically, given the context, domain-specific words like “product”, “Amazon”, “kindle”, “shop”, “cart”, and “purchase” were added. On the contrary, the word “not” was removed from the list as it is a key element for the classification task that will be discussed later on.

## 2.3 TOKENIZATION

It is well known that, in natural language processing, tokenization is a basic preprocessing step that entails dividing text into more digestible chunks, usually words or phrases. This procedure is essential for converting unstructured text into a format that is structured and easier to evaluate. For this project, we decided to perform text tokenization using the “word_tokenize” function from the NLTK package. This segmentation allows for more precise analysis and manipulation of the text data in the next processing stages.

## 2.4 STEMMING

To reduce variability and increase consistency in the data, stemming was employed to normalize words by reducing them to their root forms. This technique allowed for the consolidation of different forms of a word into a single representative form, thus minimizing redundancy. Therefore, we decided to perform stemming using the Porter Stemmer algorithm.

## 2.5 SCORE TRANSFORMATION *for classification*

As will be later discussed, one of the tasks that was decided to perform is binary classification; it becomes, therefore, crucial to manage what would be the target feature to better prepare the data for modelling. For this reason, the feature “Score”, which we recall being on a scale from 1 to 5, will be transformed into a binary form. To achieve this, all reviews with a neutral score (i.e., score of 3) were excluded. The remaining scores were therefore re-categorized into two distinct classes: negative (scores of 1 or 2) and positive (scores of 4 or 5). For simplicity, these classes were encoded as 1 for positive and 0 for negative. Moreover, as the feature “Score” was found to be unbalanced, the respective classes used as target variable were therefore balanced by down-sampling the dominant class, that is, the positive class.

## 2.6 DATA SPLITTING

Given the important number of reviews that needed to be processed, it was decided to split the cleaned and preprocessed dataset into two sets: the training set comprises of 70% of the original reviews, whilst the test sent of the remaining 30%. The splitting allowed us to train the machine learning models on a substantial portion while reserving a significant subset for unbiased evaluation for what concerns the classification task, and to perform clustering on a smaller but consistent portion avoiding computational issues. We care to notice that “Score” balancing step previously discussed was carried out only for the training set employed for classification, as the test set for this task was untouched for the purpose of keeping it as “real” as it was.

## 3. TEXT REPRESENTATION & DIMENSIONALITY REDUCTION

As previously mentioned, the Amazon reviews were written over a time period that goes from 1999 to 2012, which may imply that there could be difference in length between reviews written back in the first years of the current millennium and those written over the end and after its first decade. To address this concern, we decided to analyze the length of said reviews and, as *Figure 2* shows, it was possible to affirm that there is no significant difference in the number of words contained.

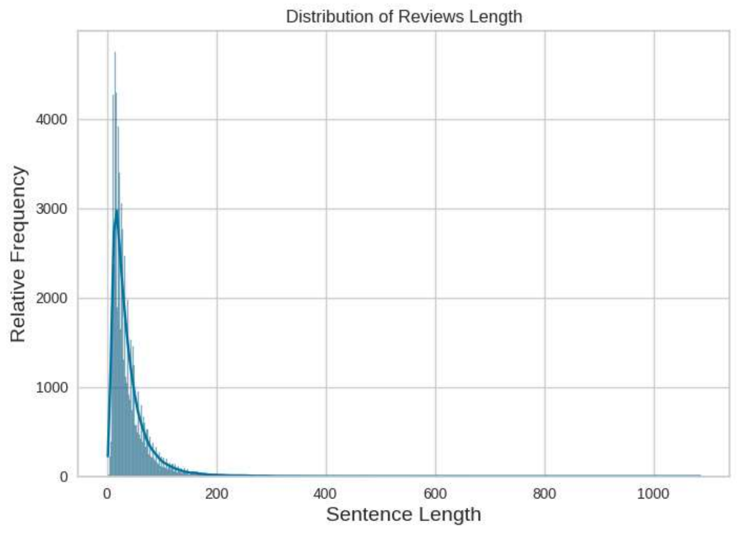


*Figure 2: Distribution of Reviews Length*

Given that the reviews in our dataset exhibit relatively uniform sentence lengths, we decided to perform text representation employing the following two techniques: Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF).

### 3.1 BAG OF WORDS

The Bag of Words model was employed counting the occurrence, i.e., the frequency, of each word in the dataset, rather than merely recording their presence or absence. The matrix produced is sparse and

contains 9.697 unique words. It emerged that the word frequency within the reviews text varies significantly, with some words not appearing at all in certain documents (hence a minimum value of 0) and others appearing very frequently (maximum value of 71).
As the number of features produced is enormous and the type of metrics is sparse, to avoid computational issue we decided to apply the Truncated Singular-Value-Decomposition (SVD) technique, with the aim of reducing the dimensionality. After said application, the number of features was reduced to 700 and the explained variance accounts for the 78% of the total variability.

### 3.2 TF-IDF

In order to have a comparison with respect to the BoW results, the TF-IDF technique was applied as well. The TF-IDF differs from the BoW matrix as it normalizes word counts by their frequency across all documents, making it a more informative measure for distinguishing important words from common ones. Like the BoW, the output is a sparse matrix identifying 9.697 unique words, where the minimum value is 0, indicating that some words are absent from some reviews, while it achieves a maximum value of 1, i.e., the highest TF-IDF score.
Also in this case, we decided to apply the Truncated Singular-Value-Decomposition (SVD) with the aim of reducing the dimensionality. After said application, the number of features was reduced to 1500 and the explained variance accounts for the 74% of the total variability.

## 4. BINARY CLASSIFICATION

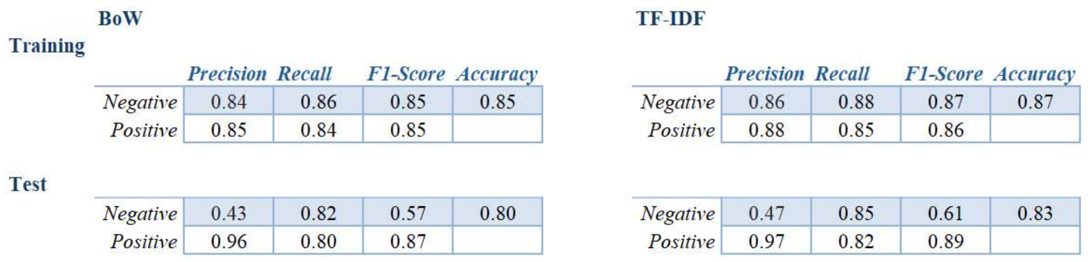
As previously mentioned, for our first task we aim to classify reviews into positive or negative, according to their score, which have been transformed into classes (0 if negative, 1 if positive). To achieve this, three different classification models were implemented:

- Light GBM
- Logistic Regression
- N-Grams Logistic Regression

In the first two cases, in order to have a comparison, both text representations previously discussed, i.e., BoW and TF-IDF, were involved in the models' fitting process.

## 4.1. LIGHT GBM

The LightGBM [3] classifier is a highly efficient and scalable gradient boosting framework. It is known to be a great option for text classification tasks since it works especially well with huge datasets and high-dimensional feature spaces. In order to make use of the advantages of each representation, we trained the LightGBM classifier on both BoW and TF-IDF features. This should have allowed us to make sure that the model could accurately capture the frequency and importance of phrases in the reviews. Although, the results obtained from the training set are captivating, those obtained from the test set are highly questionable.



**BoW**

**Training**

| | *Precision* | *Recall* | *F1-Score* | *Accuracy* |
|---|---|---|---|---|
| *Negative* | 0.84 | 0.86 | 0.85 | 0.85 |
| *Positive* | 0.85 | 0.84 | 0.85 | |

**Test**

| | | | | |
|---|---|---|---|---|
| *Negative* | 0.43 | 0.82 | 0.57 | 0.80 |
| *Positive* | 0.96 | 0.80 | 0.87 | |

**TF-IDF**

| | *Precision* | *Recall* | *F1-Score* | *Accuracy* |
|---|---|---|---|---|
| *Negative* | 0.86 | 0.88 | 0.87 | 0.87 |
| *Positive* | 0.88 | 0.85 | 0.86 | |

| | | | | |
|---|---|---|---|---|
| *Negative* | 0.47 | 0.85 | 0.61 | 0.83 |
| *Positive* | 0.97 | 0.82 | 0.89 | |

*Figure 3: LightGBM classification results.*

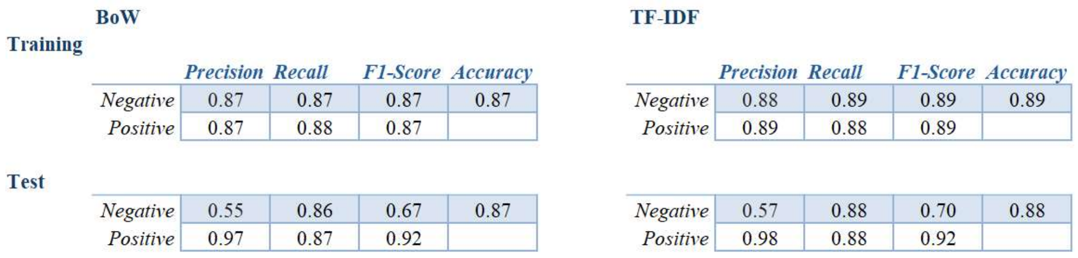
As it can be seen, the classifier fit on BoW performs well on the training data with nearly identical results for both negative and positive classes. Precision, recall, and F1-score are all around 0.84 to 0.86, and the overall accuracy is 85%, suggesting the model is well-fitted to the training data. However, the performance on the test set reveals a significant decline in classifying the negative reviews, as the precision drops to 0.43. Although the overall accuracy on the test set is 80%, the difference in class performance clearly suggests that the model does not generalize well on unseen data.

The model fit on TF-IDF performs slightly better on the training data compared to the BoW, with precision, recall, and F1-scores hovering around 0.86 to 0.88, and an overall accuracy of 87%. At the same time, on the test data, the precision and F1-scores for both classes are higher compared to BoW but still low for the negative class, where the F1-score improves to 0.61 compared to 0.57 in the BoW model.

Therefore, it can be affirmed that the model fit with TF-IDF outperforms the BoW model, particularly in its ability to generalize to the test set, but the overall performance is still low and this gives us a reason to try with other classification models.

## 4.2 LOGISTIC REGRESSION

As second approach, it was decided to shift our focus to Logistic Regression, a simpler yet effective classification model. As it was done with the LightGBM classifier, also in this case the logistic regression classifier was trained using both BoW and TF-IDF representations. The results obtained highlight slightly different patterns compared to the LightGBM classifier as it can be seen in *Figure 4*.



**Training**

**BoW**

| | *Precision* | *Recall* | *F1-Score* | *Accuracy* |
|---|---|---|---|---|
| *Negative* | 0.87 | 0.87 | 0.87 | 0.87 |
| *Positive* | 0.87 | 0.88 | 0.87 | |

**TF-IDF**

| | *Precision* | *Recall* | *F1-Score* | *Accuracy* |
|---|---|---|---|---|
| *Negative* | 0.88 | 0.89 | 0.89 | 0.89 |
| *Positive* | 0.89 | 0.88 | 0.89 | |

**Test**

**BoW**

| | *Precision* | *Recall* | *F1-Score* | *Accuracy* |
|---|---|---|---|---|
| *Negative* | 0.55 | 0.86 | 0.67 | 0.87 |
| *Positive* | 0.97 | 0.87 | 0.92 | |

**TF-IDF**

| | *Precision* | *Recall* | *F1-Score* | *Accuracy* |
|---|---|---|---|---|
| *Negative* | 0.57 | 0.88 | 0.70 | 0.88 |
| *Positive* | 0.98 | 0.88 | 0.92 | |

*Figure 4: Logistic Regression classification results.*

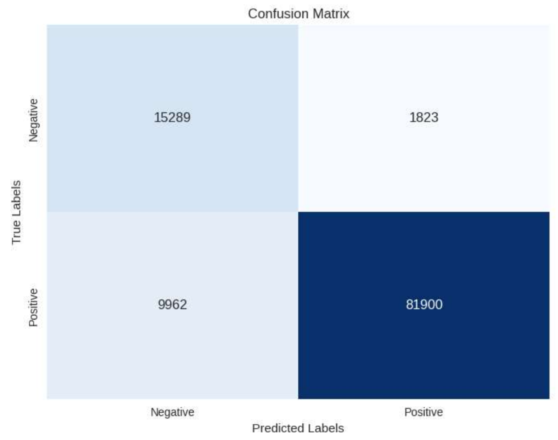
The Logistic Regression model trained with BoW shows a particularly good performance on the training data, with precision, recall, and F1-scores for both classes around 0.87, and an overall accuracy of 87%. This suggests that the model has a solid fit on the training set. On the other hand, when applied to the test set, the model's performance drops. While the positive class has a high precision (0.97) and a strong F1-score (0.92), the negative class suffers a noticeable drop in precision where the score equals 0.55. Despite an overall accuracy of 87% on the test set, this significant disparity in class performance indicates that the model struggles particularly in classifying negative reviews.

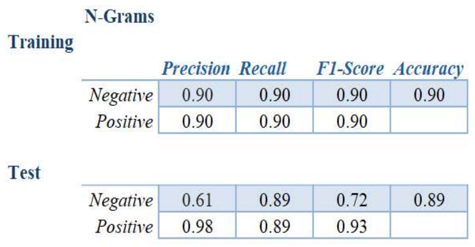
On the other hand, when using TF-IDFthe model performs slightly better than with BoW (as happened with LightGBM classifier). On the training set, precision, recall, and F1-scores for both classes improve slightly, with an accuracy of 89%, indicating a better fit. On the test set, the TF-IDF model also demonstrates better generalization than the BoW model, with the F1-score for the negative class improving to 0.70 and maintaining a high F1-score of 0.92 for the positive class. The overall test accuracy of 88% suggests that the TF-IDF representation allows the model to better balance performance across classes.

Therefore, it can be affirmed that, as with the LightGBM classifier, the Logistic Regression model fit with TF-IDF outperforms the BoW model, especially in its ability to generalize to the test set. However, while the overall performance is more balanced with TF-IDF, there still remains room for improvement, particularly in further enhancing the model's ability to accurately classify negative reviews.

## 4.3 N-GRAMS LOGISTIC REGRESSION

After evaluating the performance of LightGBM and Logistic Regression, it was decided to experiment a solution that could better capture the data nuances. In natural language processing, n-grams are used to capture patterns or relationships between words by grouping them into pairs, triples, or higher orders, depending on the value of “n”. In our case, we focus on pairs of words, in order to be able to capture instances like "not good" or "very good" which could lead to a better performance given the purpose of the task. Therefore, in this section the results obtained from the N-Grams classifier are shown. We care to notice that, as the Logistic Regression model trained with TF-IDF text representation demonstrated to outperform the other models, it was decided to fit the N-Grams using logistic regression with TF-IDF text representation.





**N-Grams**

**Training**

| | *Precision* | *Recall* | *F1-Score* | *Accuracy* |
|---|---|---|---|---|
| *Negative* | 0.90 | 0.90 | 0.90 | 0.90 |
| *Positive* | 0.90 | 0.90 | 0.90 | |

**Test**

| | *Precision* | *Recall* | *F1-Score* | *Accuracy* |
|---|---|---|---|---|
| *Negative* | 0.61 | 0.89 | 0.72 | 0.89 |
| *Positive* | 0.98 | 0.89 | 0.93 | |

*Figure 5: N-Grams classification results.*

As it can be seen from *Figure 5*, the logistic regression model, trained using n-grams combined with TF-IDF, performs effectively in classifying reviews. On the training set, the model shows a strong, balanced performance with precision, recall, and F1-scores all at 0.90, and an overall accuracy of 90%, indicating a good fit to the data. On the other hand, when evaluated on the test set, while the model still performs well with an accuracy of 89%, there is an evident difference between how it handles positive and negative reviews. For positive reviews, the model achieves high precision, as well as a F1-score. However, the precision for negative reviews drops to 0.61, suggesting that the model incorrectly classifies more negative reviews as positive. The confusion matrix also supports these observations, showing that the model correctly classifies the majority of positive reviews but struggles more with negative ones, obviously leading to a higher number of misclassifications. This model performance suggests that while the model is generally robust, it is biased toward accurately predicting positive reviews, indicating potential margin for further refinement.

## 5. CLUSTERING

The second purpose of our analysis, as mentioned in the introduction, is to explore the application of clustering techniques on the Amazon reviews to possibly uncover underlying patterns. As first goal, we aim to understand whether the reviews naturally form distinct clusters based on their scores or if there are hidden groupings independent of the given scores, but topic related. To do so, we first apply Agglomerative Hierarchical Clustering, which allows us to observe the structure and relationships within the data. Additionally, we employ K-Means clustering, to refine and possibly validate the cluster formations and their nature. We care to notice that, given the optimal results obtained from the TF-IDF, both approaches were implemented using said representation. However, to mitigate the potential impact of value distribution on the accuracy of clustering, a standardization was performed.

Before getting into details, in order to have a first glimpse into the data structure, we decided to plot, for computational limitations, a subset of 700 reviews according to their original score (1-5). To do so, the t-SNE [4] was used to reduce the high-dimensional feature set extracted from TF-IDF to two dimensions and visualize the results to gain insights into potential clustering patterns.

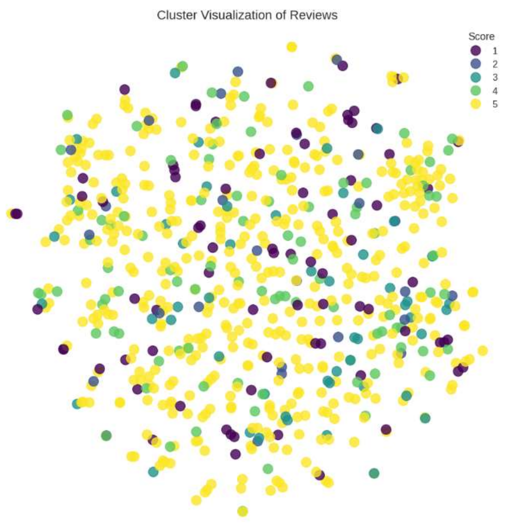


*Figure 6: reviews score plot.*

As it is possible to see from Figure 6, there is no clear separation between scores, suggesting that the reviews may not cluster distinctly by score; instead, they are mixed. However, this concern is going to be addressed in the following sections.

## 5.1 AGGLOMERATIVE HIERARCHICAL CLUSTERING

Agglomerative Hierarchical Clustering is a popular unsupervised learning technique used to group data points into clusters based on their similarity, by repeatedly merging the closest pairs of clusters. Differently to the common use of hierarchical clustering methods, which do not require specifying the number of clusters in advance, here we define the exact number of clusters that we expect based on the reviews scores, that is, five clusters.

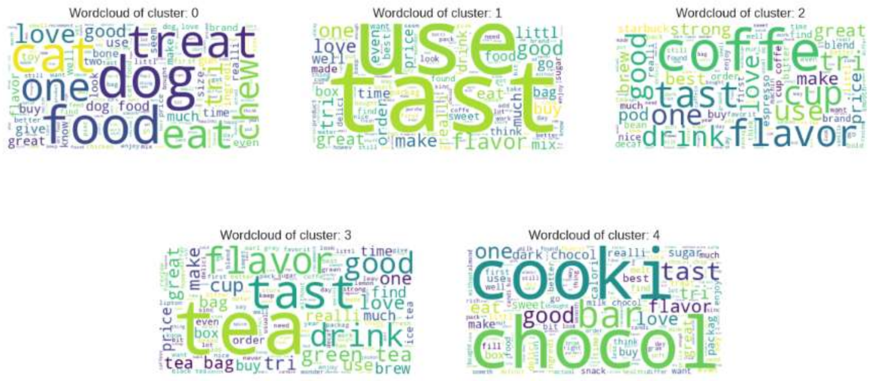


*Figure 7: Agglomerative Clusters wordclouds.*

The agglomerative clustering results indicate that the algorithm primarily grouped reviews based on content (such as dog food, coffee, and chocolate) rather than by their scores. Moreover, the clustering evaluation metrics revealed a Rand Index of 0.483, showing a considerate misalignment. The extremely low Homogeneity (0.0021) and Completeness (0.0029) scores also suggest that the clusters may contain reviews with mixed scores, and similar scores are spread across different clusters. Lastly, the Silhouette Score as well, with a value of 0.004, indicates that the clusters are poorly separated. Overall, although the clusters make sense thematically, they do not seem to align well with the review scores, suggesting that the algorithm focused more on grouping by content rather than by rating. To have a possible comparison, in the subsequent section a different approach will be discussed.

## 5.2 K-MEANS

The K-Means is a popular unsupervised machine learning algorithm used for clustering data into a predefined number of groups, i.e., clusters, and it works by partitioning the data into K clusters, which is a value specified priorly. The algorithm aims to minimize the within-cluster variance, ensuring that data points within each cluster are as similar as possible. In our specific case, as previously done, here we aim to partition the reviews into K = 5 clusters, each ideally corresponding to one of the score levels (1 to 5).

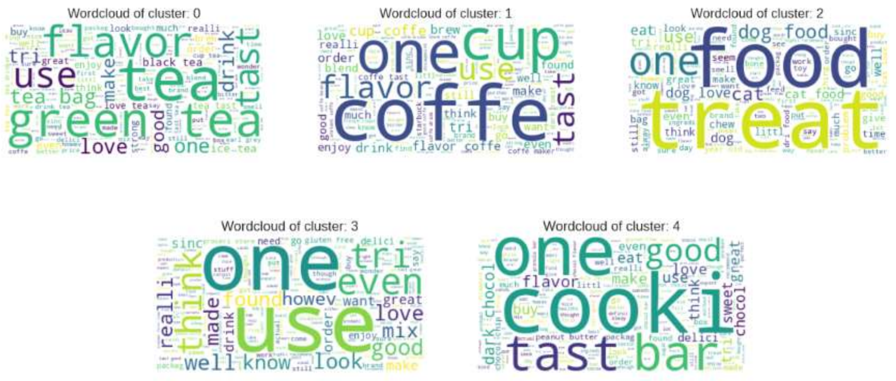


*Figure 8: K-Means with K = 5 wordclouds.*

The results obtained from the K-Means clustering show weak performance, as indicated by several key metrics. For instance, the very low Homogeneity (0.001) and Completeness (0.001) scores reveal that the clusters are not pure, meaning reviews with different scores seem to mixed within the same clusters, and reviews with the same score are spread across multiple clusters. The V-Measure of 0.001 also confirms this poor clustering quality overall. Additionally, the low Silhouette Score of 0.012 suggests that the clusters are poorly separated.

The word clouds for each cluster highlight the dominant themes within the clusters. For instance, Cluster 0 focuses on tea-related terms like "flavor" and "green," while Cluster 1 centers around coffee terms like "coffee" and "cup." Cluster 2 is dominated by dog food-related terms. However, also in this case, the presence of repetitive and overlapping terms across different clusters suggests that the algorithm grouped reviews based on content themes rather than their scores.

Overall, the clusters seem to reflect product categories, like tea, coffee, and snacks, and do not seem to be related to the review ratings. This suggests that the choice of five clusters is not optimal. For this reason, it was decided to explore different numbers of clusters with the aim of understanding a possible real clustering pattern.

In order to achieve this, the K-Means was applied according to the optimal number of clusters which was found using the Elbow Method.

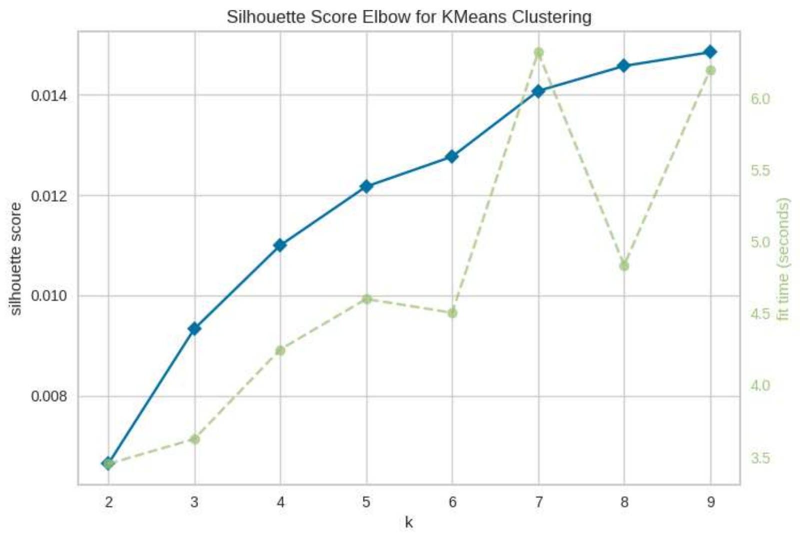


*Figure 9: optimal number of clusters.*

From the above graph it is possible to see that, as K increases, the Silhouette score gradually rises, indicating that the clustering quality improves with more clusters. This basically implies that the reviews become more distinctly grouped into meaningful clusters. Although a lower fit time would have been preferable, it is evident that the number of clusters for which the Silhouette coefficient is maximized, is given by K = 9.

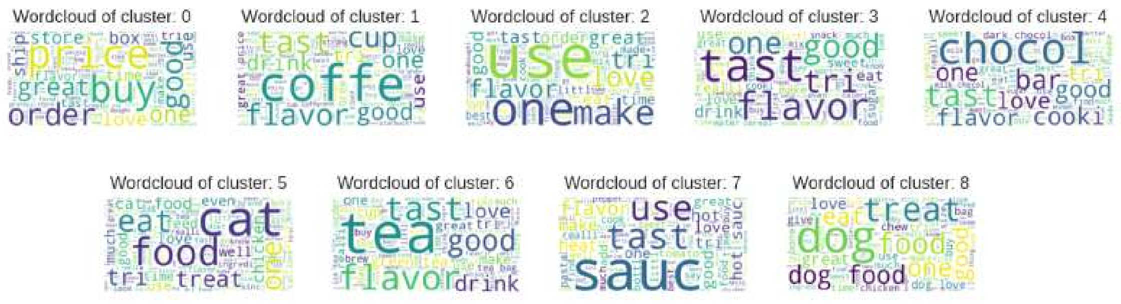


*Figure 10: K-Means with K = 9 wordclouds.*

As it can be seen from *Figure 10*, the word clouds for almost each cluster provide a clear picture of the predominant topics in the customer reviews. For instance, Cluster 1 appears to focus on reviews about coffee purchases, with words like “coffee”, “drink” and “good”, suggesting positive experiences related to buying coffee products. In Cluster 2, the common words “use” and “make” suggest that reviews in this group are about the preparation or usage of various products.

Cluster 4 clearly highlights reviews about chocolate and baked goods, as the predominant words are “chocol”, “bar” and “cooki”, while Cluster 6 emphasizes reviews about tea and drinks, as suggested by words like “tea”, “flavor”, and “drink”. Overall, it is possible to say that, even though the Silhouette coefficient has been maximized whilst remaining close to zero, the clusters identified by the K-Means algorithm provide a somehow nuanced view of the topics and sentiments expressed in the reviews.

## CONCLUSIONS

In this project, we analyzed the Amazon Fine Food Reviews dataset using advanced text preprocessing, binary classification, and clustering techniques. The approach chosen included meticulous data preparation, addressing language variability issues, such as accents and abbreviations, along with duplicated entries. The first purpose of the analysis was to explore multiple machine learning models with the aim of classify positive and negative reviews. The results revealed that, although there were important challenges in classifying negative reviews across all models, the logistic regression model with TF-IDF representation, particularly when enhanced with N-Grams, offered the most balanced and accurate performance among the models tested.

The second goal of our analysis was to apply clustering techniques to the Amazon reviews to identify underlying patterns. This involved understanding whether the reviews formed distinct clusters based on their scores or if there were hidden groupings independent of the scores but related to the topics. The results obtained from the clustering analysis demonstrated that the reviews tend to group more by content themes rather than by scores, regardless of the clustering technique used. This finding suggests that review text contains rich information about product categories, which may not align with user ratings.

## REFERENCES

[1] Kaggle: https://www.kaggle.com/

[2] NLTK Stopwrods: https://www.nltk.org/search.html?q=stopwords

[3] Light GBM: https://lightgbm.readthedocs.io/en/stable/Python-Intro.html

[4] TSNE Scikit Learn: https://scikit
learn.org/stable/modules/generated/sklearn.manifold.TSNE.html