International Burch University

INTRODUCTION TO NATURAL LANGUAGE PROCESSING

Exploratory Data Analysis for Amazon food reviews data set

Author:

Eman Hrustemović

Professor:

Amela Vatreš Mudželet and Adnan Dželilhodžić

Sarajevo, June, 2024.

Contents:

ABSTRACT

In the digital age, online reviews have become a significant factor influencing consumer behavior and market trends. This study focuses on the analysis of Amazon food reviews, a dataset comprising approximately 568,454 reviews collected over a decade. Amazon, a global online retail giant, garners a plethora of customer reviews that serve as a mirror to consumer sentiment and behavior. The objective of this project is to perform data cleaning and preprocessing, perform data analysis, applying machine learning / deep learning concepts using Python’s panda’s library on this dataset.

I aim to understand the sentiments embedded in these reviews and their correlation with variables such as product ratings and review helpfulness. The methodology involves the use of Data Mining techniques for processing and analyzing the review text. The insights derived from this study are expected to provide a deeper understanding of customer preferences in the online food retail sector, thereby contributing to strategies for enhancing customer satisfaction and engagement.

Keywords: Amazon, food reviews, consumer behavior, exploratory data analysis (EDA), pandas, Data Mining.

1. INTRODUCTION

In the contemporary digital landscape, online reviews have emerged as a crucial resource for consumers across the globe. They provide insights into the quality of products and services, influence purchasing decisions, and shape public opinion. Amazon, one of the e-commerce giants, is a platform that people use every day for online purchases. Here, they can read thousands of reviews dropped by other customers about their desired products. These reviews provide valuable opinions about a product such as its property, quality, and recommendations, which help the purchasers to understand almost every detail of a product. This is not only beneficial for consumers but also helps sellers who are manufacturing their own products to understand the consumers and their needs better.

However, as the number of reviews available for a product grows, it is becoming more difficult for a potential consumer to make a good decision on whether to buy the product. Different opinions about the same product on one hand and ambiguous reviews on the other hand makes customers more confused to get the right decision.

This project centers around the analysis of food reviews on Amazon, a comprehensive dataset that spans over a decade and encompasses approximately 568,454 reviews. Amazon, being one of the world's largest online retailers, amasses a vast array of customer reviews. These reviews serve not only as feedback on products but also mirror consumer behaviour and sentiment. By scrutinizing these reviews, we can extract valuable insights into customer preferences and trends.

The aim of this project is to conduct an exploratory data analysis (EDA) of the Amazon food reviews dataset using Python's libraries. I strive to comprehend the sentiments expressed in these reviews and their correlation with factors such as the product rating and the helpfulness of the review. To achieve this, we will employ Data Mining (DM) techniques to process and analyze the text of the reviews. This endeavour will enhance our understanding of consumer sentiment in the online food retail sector and potentially offer strategies for improving customer satisfaction and engagement. The use of EDA and other techniques will allow me to effectively handle and analyze this large dataset.

2. THEORETICAL BACKGROUND/LITHERATURE REVIEW

2.1. Data mining

Data Mining, also known as Knowledge Discovery in Databases (KDD), is a process of automatically discovering patterns, relationships, and insights from large datasets. Its goal is to extract valuable knowledge and patterns from complex data, often using statistical and machine learning techniques. Data mining involves several steps, including data selection, preprocessing, pattern evaluation, and knowledge representation. By applying data mining techniques, organizations can gain valuable insights into customer behavior, market trends, and business operations, enabling them to make data-driven decisions and improve their overall performance.

2.2. Exploratory data analysis

Exploratory Data Analysis (EDA) is an approach to analyzing datasets to summarize their main characteristics, often with visual methods. It’s a crucial step before formal modelling or data analysis, allowing one to understand the data better, often with visual methods. EDA helps to reveal the underlying structure of the data, providing a powerful tool for summarizing and visualizing the important characteristics and relationships within the dataset. It uses visual methods to highlight important details before diving into detailed analysis.

2.3. Pandas

Pandas is a Python library for data manipulation and analysis, providing data structures like DataFrames for organizing and exploring structured data. It simplifies tasks such as data cleaning, transformation, and statistical analysis, making it a powerful tool for working with tabular data. Pandas is widely used in data science and analytics due to its efficiency and ease of use.

2.4. MATPOLIB.PYPLOT

Matplotlib.pyplot is a Python library for creating visualizations, particularly plots and charts. It provides a simple interface for generating various types of plots, enabling users to visualize data in a clear and concise manner. With functionalities like line plots, scatter plots, and histograms, pyplot is widely used in data analysis and scientific computing. It works seamlessly with NumPy arrays and is an essential tool for exploring and presenting data visually.

2.5. Related work

The authors in [6] recognized the importance of preprocessing categorical data for machine learning algorithms. Given that a significant portion of the dataset consists of categorical data, encoding techniques were applied to convert these categorical features into a format suitable for modeling.

The primary algorithm chosen for data modeling was the Random Forest Classifier. This supervised learning algorithm constructs an ensemble of decision trees using a technique called bagging. The bagging method involves training multiple decision trees independently and combining their predictions to enhance the overall accuracy and robustness of the model.

The Random Forest Classifier is particularly well-suited for ensemble learning. By aggregating the predictions of multiple decision trees, each trained on a subset of the data, the model reduces overfitting and provides a more reliable prediction, contributing to improved overall model performance.

They also did data binning. This means they grouped similar numbers together to make the data easier to understand. By doing this, they could handle extreme values better and reduce confusing fluctuations in the data. This also helped them see patterns more clearly.

The grouped data was then smoothly added to their overall plan, making it easier to build a helpful model and understand the dataset better. Overall, data binning was a key step that made the Netflix dataset analysis more organized and understandable.

In summary, the analysis on the Netflix dataset involved preprocessing steps, including the encoding of categorical data, followed by the application of the Random Forest Classifier algorithm for ensemble learning. This approach aims to enhance predictive performance and generate valuable insights from the Netflix dataset.

The other project that we analysed [7] Data is from the "ZOMATO BANGALORE RESTAURANTS" dataset on Kaggle, which has details for over 51,000 restaurants in Bangalore. They looked at factors like online orders, ratings, and cuisines.

The project analyzed a dataset of Bangalore restaurants, using geospatial analysis to map their locations in Python. Most restaurants clustered in central Bangalore. They employed the "geopy" function for latitude and longitude, creating a heatmap.

For sentiment analysis, they used TextBlob to categorize feedback as positive, negative, or neutral. The project included a Wordcloud visualization, highlighting frequently mentioned words in customer feedback, aiding in identifying positive and negative sentiments. Wordclouds were created for various restaurant types like Casual Dining and Cafe. The combination of geospatial and sentiment analysis provided insights for restaurant location planning and customer satisfaction improvement.

In the project, the focus was on text preprocessing and analysis using Data Mining (DM) techniques. They broke down paragraphs into sentences, leveraging NLTK's sent_tokenizer for encoding and analysis. Tokenization was employed to identify syntactic units in the corpus, and parts of speech tagging helped distinguish the functions of words in sentences.

They removed stop words, such as "a" and "the," to concentrate on more meaningful content, and performed text normalization to reduce features. The results were presented using Latent Dirichlet Allocation (LDA) on "Gordon Ramsay BurGR." Key findings included the distribution of ratings, prominent review locations, top categories, and insights from user reviews, check-ins, and restaurant statistics. Visualization techniques were utilized to highlight positive and negative words, and LDA showcased the most frequent topics in a visual format.

This comprehensive approach to text preprocessing and analysis allowed for a deeper understanding of the data, revealing patterns and insights that contribute to a more informed interpretation of the information within the text dataset.

Furthermore, the project utilized visualization techniques to depict both positive and negative sentiments through word clouds. The Latent Dirichlet Allocation (LDA) method was applied to identify and present the most frequent topics in a visual format. This comprehensive approach not only processed text effectively but also yielded actionable insights for understanding user sentiments, preferences, and engagement patterns.

3. DATASET

The dataset for this project is a comprehensive collection of fine food reviews from Amazon. The data spans a period of over a decade, capturing all reviews up to October 2012. This dataset is particularly rich, providing a wide range of information for each review.

Time Span: The dataset covers a substantial time frame, with reviews dating from October 1999 to October 2012. This allows for the analysis of trends and patterns over time.

Volume: The dataset is quite large, containing approximately 568,454 reviews.

Users: The reviews are provided by 256,059 unique users. This diversity can help ensure a more representative understanding of consumer sentiments.

Products: The reviews encompass 74,258 unique products, offering a broad spectrum of items for analysis.

Active Users: Dataset includes 260 users who have given more than 50 reviews each.

DATA INCLUDES:

- Reviews from Oct 1997 - Oct 2012
- 568,454 reviews 3. 256,059 users
- 74,258 products
- 260 users with > 50 reviews

ATTRIBUTE INFORMATION

- Id
- ProductId - unique identifier for the product
- UserId - unqiue identifier for the user
- ProfileName
- HelpfulnessNumerator - number of users who found the review helpful
- HelpfulnessDenominator - number of users who indicated whether they found the review helpful or no
- Score - rating between 1 and 5
- Time - timestamp for the review
- Summary - summary of the review
- Text - text of the review

4. METHODOLOGY

- DATA LOADING AND OVERVIEW

The project begins with loading a dataset from a CSV file containing reviews. Basic information about the DataFrame, such as data types and the first 1000 rows, is displayed for initial exploration. Missing values are identified and addressed.

- DATA CLEANING

Duplicate rows are checked and removed from the dataset. Additionally, rows with missing values in specific columns ('ProfileName' and 'Summary') are dropped to ensure data integrity.

- TEXT CLEANING

Text data in the 'Text' column undergoes cleaning using BeautifulSoup and regular expressions. This process involves removing HTML tags and non-alphanumeric characters, creating a cleaner text representation.

- WORD FREQUENCY ANALYSIS

A subset of the dataset is analyzed for word frequency using the CountVectorizer from scikit-learn. The top N most frequent words are identified and displayed in a bar chart.

- TEMPORAL ANALYSIS

The 'Time' column is converted to datetime, and a new 'Year' column is created. The distribution of reviews over the years is visualized using a histogram.

- PRODUCT ANALYSIS

The most-reviewed products and their review counts are identified. Additionally, products with the highest and lowest average scores are determined and presented.

- REVIEW LENGTH ANALYSIS

The length of each review is calculated and visualized using a histogram, providing insights into the distribution of review lengths.

- SCORE ANALYSIS

The distribution of review scores is visualized, and the correlation between review length and score is calculated and displayed.

- USER ANALYSIS

User review counts are analyzed, and the top users with the most reviews are identified and visualized.

- SLANG WORDS ANALYSIS

A list of slang words is defined, and a new column is created to identify reviews containing these words. The most common slang words are then counted and presented.

- WORD CLOUD GENERATION

A word cloud is generated to visually represent the most frequent words in the reviews.

- TOP REVIEWERS OVER TIME ANALYSIS

In this analysis, I visualized the distribution of review counts over time for the top reviewers, highlighting the most prolific contributors to the dataset.

- HELPFULNESS ANALYSIS

In this analysis, I explored the distribution of scores and review text lengths to compare the characteristics of reviews classified as highly helpful and less helpful based on a predefined helpfulness threshold.

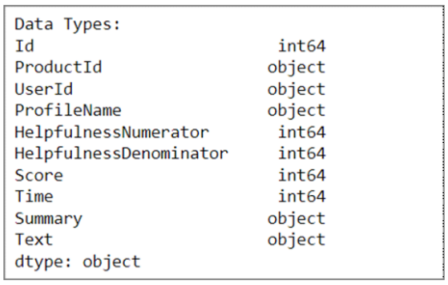
5. RESULTS

5.1. Data overview

The dataset comprises a total of 568,401 reviews. Cumulatively, the reviews contain 46,456,083 words, reflecting the richness and diversity of textual content.

In the initial exploration of the dataset, it was found that the data comprises various data types across its columns. The 'Id,' 'HelpfulnessNumerator,' 'HelpfulnessDenominator,' 'Score,' and 'Time' columns are of integer type, while 'ProductId,' 'UserId,' 'ProfileName,' 'Summary,' and 'Text' columns are of object type.

This information was obtained by loading the dataset and using the info() method in Python. It provides a foundational understanding of the dataset's structure for subsequent analysis.



```
Data Types:
Id                          int64
ProductId                  object
UserId                     object
ProfileName                object
HelpfulnessNumerator        int64
HelpfulnessDenominator      int64
Score                       int64
Time                        int64
Summary                    object
Text                       object
dtype: object
```

Fig.1.Data types

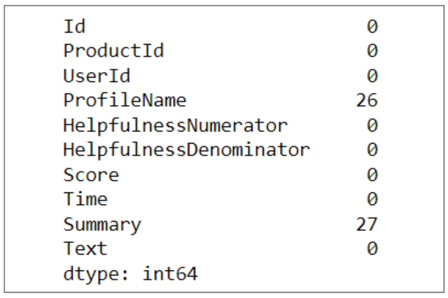
5.2. Data cleaning

In the data cleaning process, steps were taken to ensure the quality and integrity of the dataset. Duplicate rows were identified and removed to eliminate redundancy and maintain the uniqueness of each record.

This step is crucial for preventing skewed analyses and ensuring accurate insights. Furthermore, to address missing values, specific attention was given to columns such as 'ProfileName' and 'Summary.' Rows containing null values in these columns were dropped from the dataset.

Before the data cleaning process, the dataset may have contained duplicate rows and entries with missing values in critical columns. These elements could potentially introduce inaccuracies and inconsistencies in the analysis.

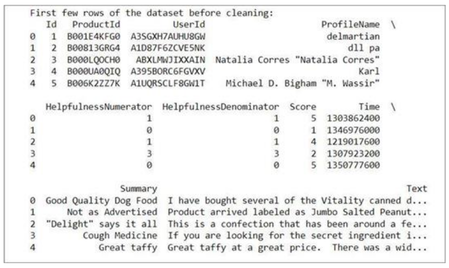
The subsequent data cleaning steps were undertaken to address these issues, ensuring a refined and reliable dataset for further investigation.



```
Id                          0
ProductId                   0
UserId                      0
ProfileName                26
HelpfulnessNumerator        0
HelpfulnessDenominator      0
Score                       0
Time                        0
Summary                    27
Text                        0
dtype: int64
```

Fig.2. Dataset rows before cleaning

5.3. Text cleaning



```
First few rows of the dataset before cleaning:
   Id   ProductId          UserId                      ProfileName  \
0   1  B001E4KFG0  A3SGXH7AUHU8GW                       delmartian
1   2  B00813GRG4  A1D87F6ZCVE5NK                           dll pa
2   3  B000LQOCH0   ABXLMWJIXXAIN  Natalia Corres "Natalia Corres"
3   4  B000UA0QIQ  A395BORC6FGVXV                             Karl
4   5  B006K2ZZ7K  A1UQRSCLF8GW1T    Michael D. Bigham "M. Wassir"

   HelpfulnessNumerator  HelpfulnessDenominator  Score        Time  \
0                     1                       1      5  1303862400
1                     0                       0      1  1346976000
2                     1                       1      4  1219017600
3                     3                       3      2  1307923200
4                     0                       0      5  1350777600

                 Summary                                               Text
0  Good Quality Dog Food  I have bought several of the Vitality canned d...
1      Not as Advertised  Product arrived labeled as Jumbo Salted Peanut...
2  "Delight" says it all  This is a confection that has been around a fe...
3         Cough Medicine  If you are looking for the secret ingredient i...
4            Great taffy  Great taffy at a great price.  There was a wid...
```

Fig.3. Dataset rows before text cleaning

After the data cleaning process, duplicate rows were identified and removed from the dataset. Additionally, rows with missing values in specific columns, such as 'ProfileName' and 'Summary', were systematically dropped. This cleaning ensures that the dataset is now free from redundancies and maintains a high level of data integrity, laying a solid foundation for subsequent analyses.

After performing text cleaning on the 'Text' column, several preprocessing steps were applied to enhance the quality of the textual data. The process involved removing HTML tags using BeautifulSoup and eliminating non-alphanumeric characters through regular expressions. The result is a refined 'Text' column with improved cleanliness and uniformity, facilitating more accurate and meaningful text analysis.

5.4. Word frequency analysis

Word frequency analysis was conducted to identify the most common words in the reviews. The 'Text' column was processed using a CountVectorizer, which transformed the text data into a matrix of word counts. The resulting word counts were then aggregated and analyzed to create a DataFrame showcasing the top N most frequent words.

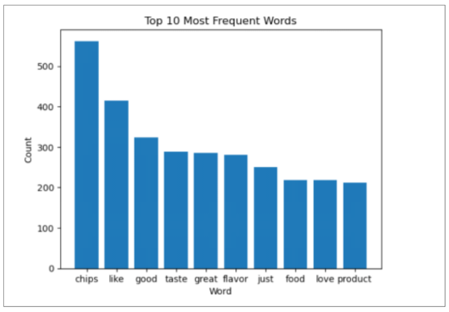


Fig.4. Top 10 most frequent words

A bar plot was generated to visually represent the distribution, providing insight into the prevalent terms within the dataset. This analysis contributes to a better understanding of the key words that appear frequently in the reviews.

In this analysis, the term 'chips' emerged as the most frequently occurring word. The height of the bar for 'chips' indicates the frequency of its appearance in the reviews.

This information is valuable for gaining insights into the common themes or topics discussed in the dataset, with 'chips' being a prominent keyword among the reviews.

5.5. Temporal analysis

In this analysis I visualized the distribution of reviews over the years using a histogram to identify trends in review activity.

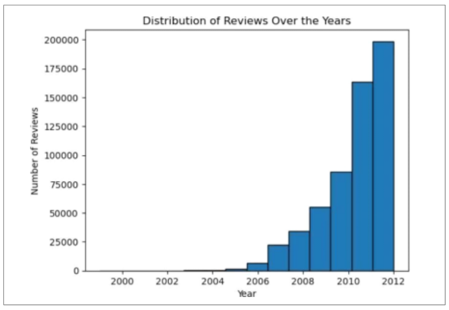


Fig.5. Distribution of reviews over the years

Enhancing the presentation of the time distribution in the code, we can observe distinct periods. Between 2010 and 2012, there were approximately 200,000 reviews, followed by a slightly reduced volume of around 175,000 reviews between 2012 and 2014.

Further back in time, specifically from 2008 to 2010, there were about 100,000 reviews. Lastly, in the earliest period between 2006 and 2008, the review count decreased to approximately 75,000. This visualization provides a clearer understanding of the evolving review trends over these specified time intervals.

5.6. Product analysis

The most-reviewed products and their review counts are identified. Additionally, products with the highest and lowest average scores are determined and presented.

The bar chart illustrates the distribution of reviews for the top products, emphasizing the most-reviewed products based on the number of reviews. The horizontal bar plot provides a clear comparison, enhancing the visual impact of the top products and their respective review counts.

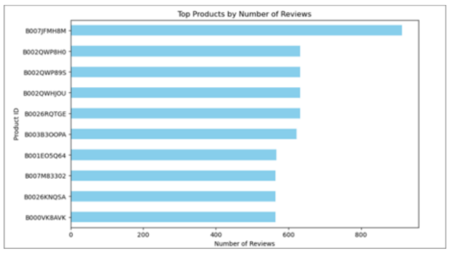


Fig.6. Top products by number of reviews

Horizontal bar plot visualizes the top products based on the number of reviews. Notably, the product with the identifier "B007JFMH8M" boasts an impressive count of over 800 reviews, making it the highest-reviewed product.

Following closely, another product with the identifier "B002QWP8HO" has garnered more than 600 reviews, securing its position as one of the top-reviewed products.

This graphical representation offers a quick and effective overview of the most- reviewed products in the dataset, with attention-grabbing insights into their respective review counts.

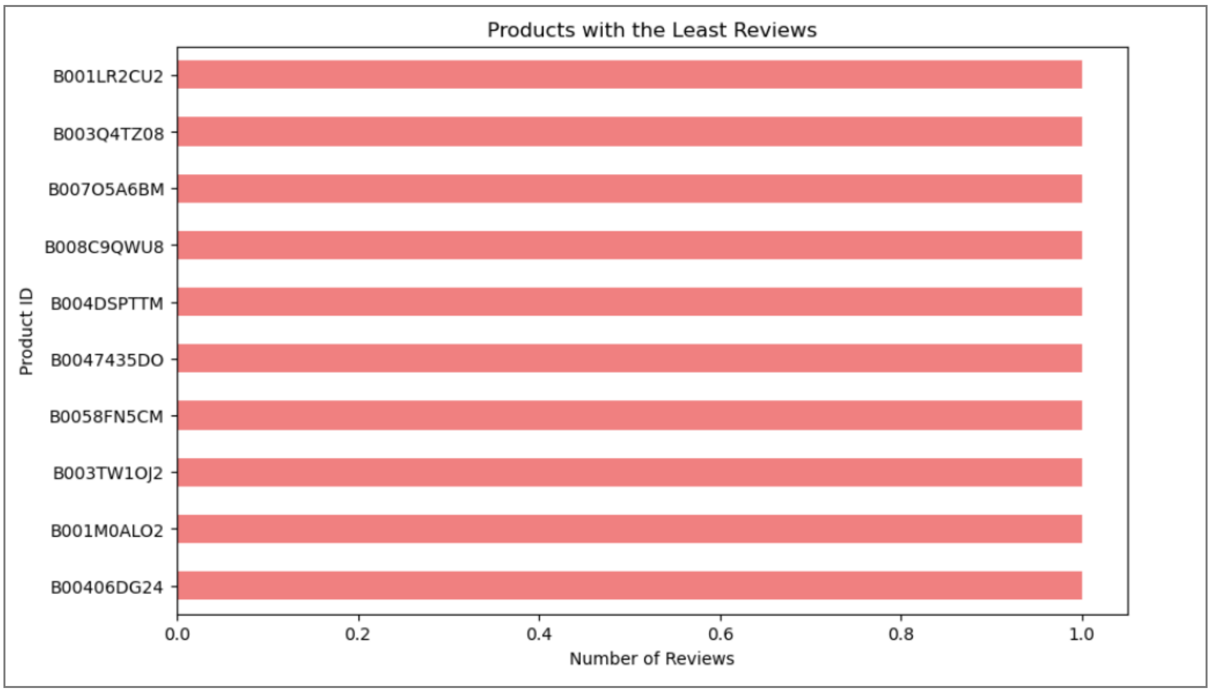
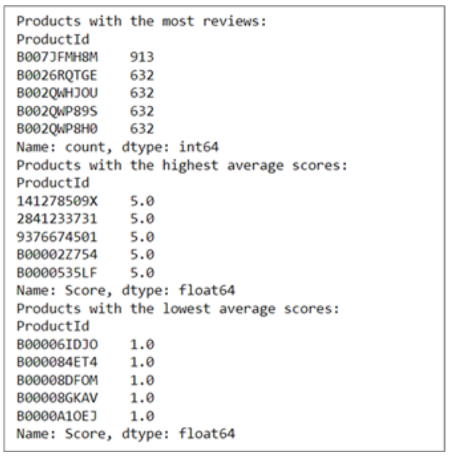


Fig.7. Products with the least reviews

The bar plot illustrates products with the least reviews, providing a visual representation of those at the lower end of the review count spectrum.

These products, identifiable by their respective product IDs, have received fewer reviews compared to others in the dataset.



```
Products with the most reviews:
ProductId
B007JFMH8M    913
B0026RQTGE    632
B002QWHJOU    632
B002QWP89S    632
B002QWP8H0    632
Name: count, dtype: int64
Products with the highest average scores:
ProductId
141278509X    5.0
2841233731    5.0
9376674501    5.0
B00002Z754    5.0
B0000535LF    5.0
Name: Score, dtype: float64
Products with the lowest average scores:
ProductId
B00006IDJO    1.0
B000084ET4    1.0
B00008DFOM    1.0
B00008GKAV    1.0
B0000A1OEJ    1.0
Name: Score, dtype: float64
```

Fig.8. Analysis of the product

Firstly, it identifies which products have the most reviews, and it turns out 'B007JFMH8M' is the top dog with a whopping 913 reviews. This suggests it's quite popular among customers. Moving on, the analysis checks out the average scores for each product. Some products, like '141278509X', '2841233731', and '9376674501', are standouts with perfect scores of 5.0, indicating customers are happy with them.

On the flip side, products such as 'B00006IDJO' and 'B00008GKAV' have the lowest average scores of 1.0, signaling potential areas for improvement.

This detailed look at product reviews provides a clear picture of what's popular and how satisfied customers are, offering valuable insights for anyone interested in understanding customer sentiments.

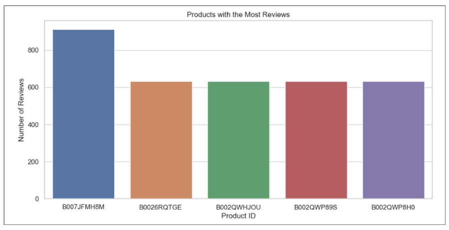


Fig.9.Products with the most review

This analysis provides insights into product popularity based on the number of reviews and evaluates customer satisfaction by identifying products with the highest and lowest average scores.

Through graphical representation, it becomes evident which products have garnered the most attention in terms of reviews.

5.7. Review length analysis

In my initial analysis of the first 1000 reviews, I observed a diverse range in review lengths. The longest review was 5276 characters, offering a detailed perspective, while the shortest was a concise 57 characters.

Extending my investigation to the entire dataset, I found the longest review to be 21079 characters and the shortest to be just 12 characters. This variation highlights the dynamic nature of review lengths across the dataset.

I found that the longest review is 21079 characters and the shortest just 12 characters. This variation highlights the dynamic nature of review lengths across the dataset.

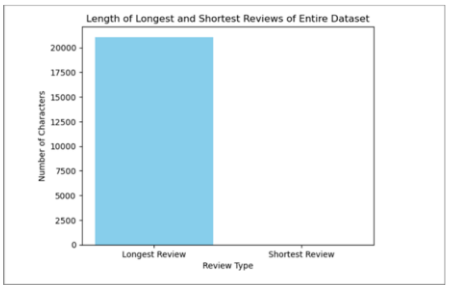


Fig.10 Length of Longest and Shortest Reviews of Entire Dataset

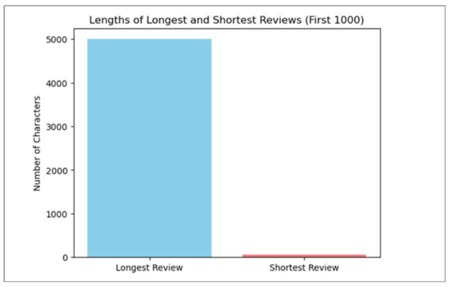


Fig.11. Length of the Longest and Shortest Reviews of First 1000 Rows

In the analysis of the first 1000 rows of the dataset, it was observed that the shortest review consists of 57 characters, conveying a relatively concise sentiment.

The review expresses a neutral opinion, stating that the product is deemed acceptable but not compelling enough to warrant future purchases.

This succinct feedback implies a moderate level of satisfaction without strong enthusiasm.

The dataset's character count analysis indicates variability in review lengths, with the briefest review providing a glimpse into concise and straightforward customer sentiments within the initial subset of data

5.8. Score analysis

Why do we need Distribution of Scores?Understanding the distribution of scores provides valuable insights into the overall sentiment of customer reviews. It helps identify whether many reviews are positive, neutral, or negative, offering a comprehensive overview of customer satisfaction.

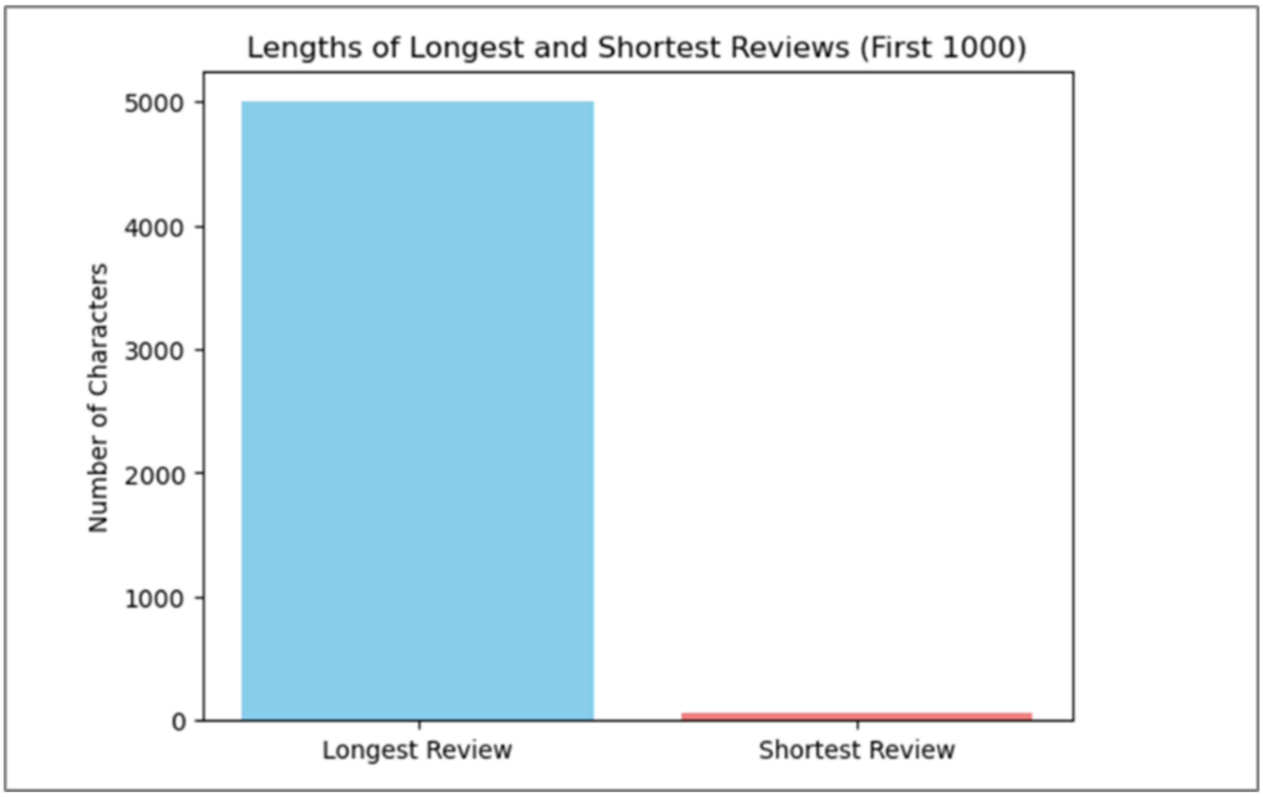


Fig.12. Length of Longest and Shortest Reviews of Entire Dataset

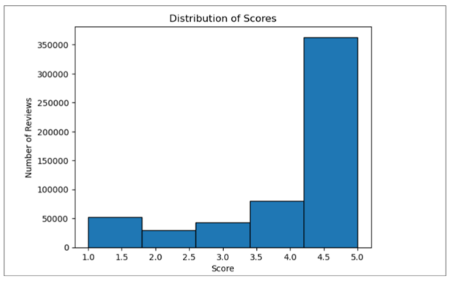
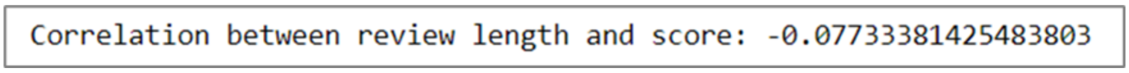


Fig.13. Distribution of scores

The score analysis reveals the distribution of scores within the dataset, illustrated by a histogram. Approximately 350,000 reviews are concentrated between scores just shy of

4.5 and a perfect score of 5.0, indicating a prevalence of high scores and generally positive reviews. In contrast, there are around 50,000 reviews with scores below 1.5, suggesting a smaller but notable proportion of lower-scoring or potentially negative reviews. Additionally, the correlation between the length of reviews and their scores was calculated. The correlation coefficient of approximately -0.08 indicates a very weak negative correlation.

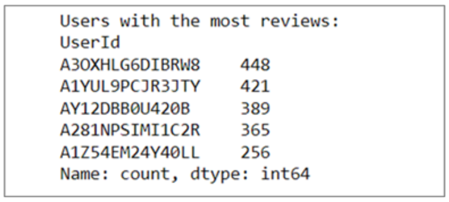


```
Correlation between review length and score: -0.07733381425483803
```

In summary, the correlation coefficient indicates a weak negative correlation between review length and score, implying that longer reviews may slightly tend to have lower scores, but the relationship is not strong.

5.9. User analysis

This user analysis focuses on understanding user engagement with the dataset by examining the distribution of reviews contributed by each user. The first part counts the number of reviews for each user, revealing the users who are most active in terms of providing reviews.



```
Users with the most reviews:
UserId
A3OXHLG6DIBRW8    448
A1YUL9PCJR3JTY    421
AY12DBB0U420B     389
A281NPSIMI1C2R    365
A1Z54EM24Y40LL    256
Name: count, dtype: int64
```

Fig.14.Users with the most reviews analysis

The output lists the top five users with the highest number of reviews in a dataset. Each user is identified by a unique UserId, and the corresponding count represents the total number of reviews contributed by that user. The information highlights the most prolific contributors to the reviews, offering insights into user engagement patterns and potential influencers within the dataset. Users with higher review counts may have a significant impact on the overall sentiment and feedback captured in the data, making this analysis valuable for understanding user behavior and preferences.

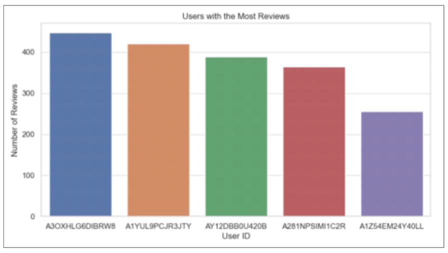
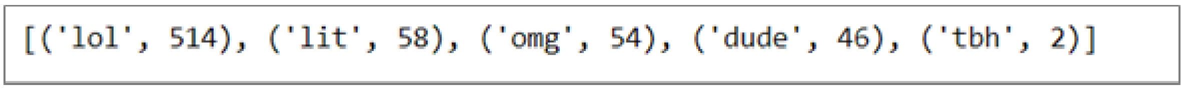


Fig.15. Users with most reviewws

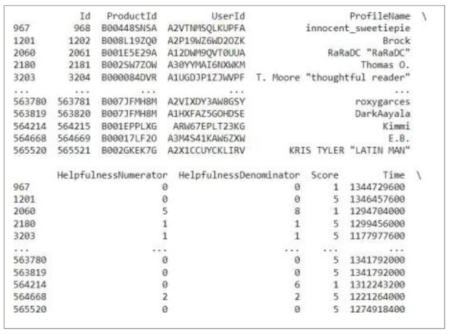
A bar plot visualizes the users with the highest number of reviews, where the x-axis represents user IDs, the y-axis represents the number of reviews, and each bar indicates a user's review count. In Fig.15 we can see that the plot illustrates the distribution of user reviews, with UserId "A3OXHLG6DIBRW8" standing out as the top contributor with more than 448 reviews.

5.10. Slang word analysis

This analysis provides insights into the prevalence of slang words within the reviews, highlighting the frequently used informal expressions. By quantifying and displaying the most common slang terms, the analysis aims to uncover patterns of informal language usage in the dataset, offering a unique perspective on the linguistic characteristics of the reviews.



```
[('lol', 514), ('lit', 58), ('omg', 54), ('dude', 46), ('tbh', 2)]
```

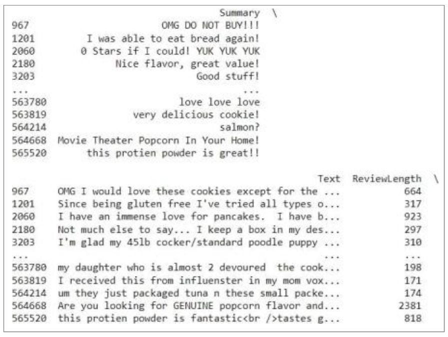


```
             Id   ProductId          UserId                    ProfileName  \
967         968  B00448SNSA  A2VTNMSQLKUPFA            innocent_sweetiepie
1201       1202  B008L19ZQ0  A2P19WZ6WD2OZK                          Brock
2060       2061  B001E5E29A  A12DWM9QVT0UUA                RaRaDC "RaRaDC"
2180       2181  B002SW7ZOW  A30YYMAI6NXWKM                     Thomas O.
3203       3204  B000084DVR  A1UGDJP1ZJWVPF  T. Moore "thoughtful reader"
...         ...         ...             ...                           ...
563780  563781  B007JFMH8M  A2VIXDY3AW8GSY                     roxygarces
563819  563820  B007JFMH8M  A1HXFAZ5GOHDSE                     DarkAayala
564214  564215  B001EPPLXG   ARW67EPLT23KG                          Kimmi
564668  564669  B00017LF2O  A3M4S41KAW6ZXW                          E.B.
565520  565521  B002GKEK7G  A2X1CCUYCKLIRV       KRIS TYLER "LATIN MAN"

        HelpfulnessNumerator  HelpfulnessDenominator  Score        Time  \
967                        0                       0      1  1344729600
1201                       0                       0      5  1346457600
2060                       5                       8      1  1294704000
2180                       1                       1      5  1299456000
3203                       1                       1      5  1177977600
...                      ...                     ...    ...         ...
563780                     0                       0      5  1341792000
563819                     0                       0      5  1341792000
564214                     0                       6      1  1312243200
564668                     2                       2      5  1221264000
565520                     0                       0      5  1274918400
```

Fig.16.

The data shown in Fig.16 is part of a bigger collection, and it is all about product reviews. Each row is like a card with details about one review. The "Id" is like a special number for each review, and "ProductId" is the product's ID.

"UserId" tells us who wrote the review, and "ProfileName" is their chosen name. The review's helpfulness is in "HelpfulnessNumerator" and "HelpfulnessDenominator," showing how many people found it useful. "Score" is the user's rating from 1 to 5, and "Time" says when the review was written.

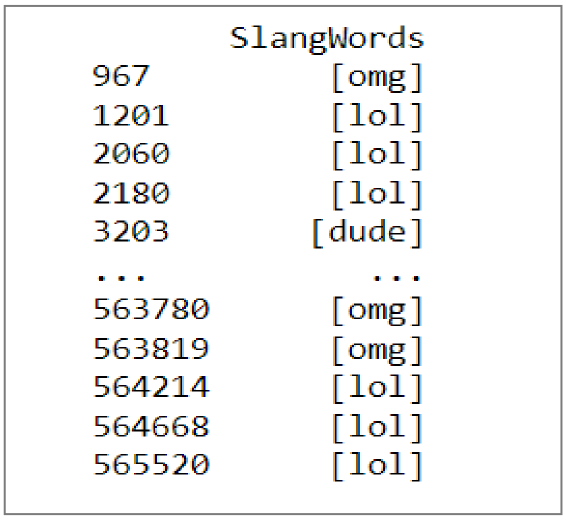


```
                                    Summary  \
967                       OMG DO NOT BUY!!!
1201             I was able to eat bread again!
2060            0 Stars if I could! YUK YUK YUK
2180                 Nice flavor, great value!
3203                               Good stuff!
...                                        ...
563780                        love love love
563819                  very delicious cookie!
564214                                 salmon?
564668  Movie Theater Popcorn In Your Home!
565520          this protien powder is great!!

                                                     Text  ReviewLength  \
967     OMG I would love these cookies except for the ...           664
1201    Since being gluten free I've tried all types o...           317
2060    I have an immense love for pancakes.  I have b...           923
2180    Not much else to say... I keep a box in my des...           297
3203    I'm glad my 45lb cocker/standard poodle puppy ...           310
...                                                   ...           ...
563780  my daughter who is almost 2 devoured  the cook...           198
563819  I received this from influenster in my mom vox...           171
564214  um they just packaged tuna n these small packe...           174
564668  Are you looking for GENUINE popcorn flavor and...          2381
565520  this protien powder is fantastic<br />tastes g...           818
```

Fig.17.

This section presents details on reviews, emphasizing the use of slang words. Each row includes a concise "Summary" offering a quick glimpse into the reviewer's main point, and a more detailed "Text" column providing the complete review content. The "ReviewLength" column quantifies the length of each review.

The inclusion of specific reviews, like *"OMG DO NOT BUY!!,"* suggests a focus on those incorporating slang. The analysis aims to understand the extent of informal language usage, potentially revealing patterns in expression.



```
            SlangWords
967              [omg]
1201             [lol]
2060             [lol]
2180             [lol]
3203            [dude]
...                ...
563780           [omg]
563819           [omg]
564214           [lol]
564668           [lol]
565520           [lol]
```

Fig.18.Reviews with slang words

The inclusion of diverse slang words, such as "omg," "lol," and "dude," underscores the variety of informal expressions captured.

The focused nature of this subset, with 651 reviews identified, facilitates a nuanced examination of linguistic patterns and the prevalence of slang in user feedback. By delving into these reviews, analysts gain valuable insights into the informal language landscape within the dataset, contributing to a more comprehensive understanding of user sentiment and communication styles.

This DataFrame subset displays reviews where specific slang words are present. The "SlangWords" column indicates the detected slang words for each corresponding review.

For example:

-In the review at index 967, the slang word "omg" is present.

-In the review at index 1201, the slang word "lol" is present.

-In the review at index 2060, the slang word "lol" is present.

-In the review at index 2180, the slang word "lol" is present.

-In the review at index 3203, the slang word "dude" is present.

The subset provides a focused view of reviews containing slang language, aiding in the exploration of informal expressions and linguistic patterns within the dataset. In total, there are 651 such reviews identified in this analysis.

5.11. Word cloud generator

The word cloud generated from the reviews visually encapsulates the most frequent words. Larger words in the cloud denote higher frequencies, offering an intuitive insight into the predominant topics and expressions across the reviews.

The larger the word in the cloud, the more frequently it appears in the reviews. This intuitive visual approach allows for immediate recognition of the predominant topics and expressions that characterize the dataset.

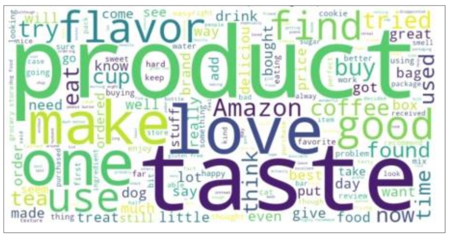


Fig.19. Word cloud

The word cloud generated from the reviews vividly displays the most frequently used words, with larger words indicating higher frequencies. In this visual representation, the words "product" and "taste" stand out prominently, suggesting their frequent occurrence in the reviews.

Following closely are words like "love," "make," "one," and "good," each contributing to the overall thematic landscape of the dataset. This intuitive visual approach enables an immediate recognition of the predominant topics and expressions that characterize the sentiments and discussions within the reviews.

5.12. Top reviewers over time analysis

The "Top Reviewers Over Time Analysis" examines the behavior of the most active reviewers in a product reviews dataset. It identifies the top 10 contributors and tracks how their review submissions change month by month over different years.

The analysis employs a colorful bar chart where each bar represents a reviewer, and its height shows how many reviews they submitted in a specific month and year.

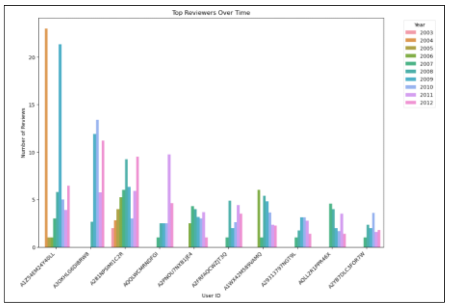


Fig.20.Top viewrs overtime

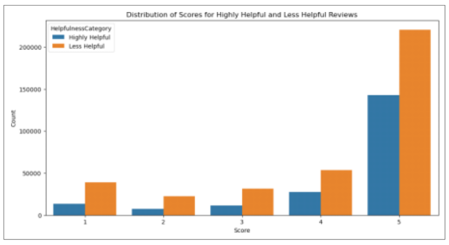
By looking at the chart, we can easily see who the most influential reviewers are and understand when their contributions peak or decline. This exploration helps us grasp the evolving engagement patterns of key reviewers, providing insights into the dynamics of their involvement with the review platform over time.

In 2004, user *A1Z54EM24Y40LL* emerged as the most prolific reviewer, contributing the highest number of reviews compared to other users. This user continued their active reviewing trend in 2009, maintaining a significant presence in generating reviews during that year as well. Conversely, in 2005 and 2006, user *A1Z54EM24Y40LL* had a notably reduced reviewing activity, making fewer contributions during these years compared to their more active periods.

5.13. Helpfulness analysis

In this study, I calculated how helpful each review is and categorized them as either 'Highly Helpful' or 'Less Helpful' based on a threshold we set. Then I made a chart using seaborn to compare the scores of reviews in these two categories, showing how scores are distributed. Additionally, I printed some numbers to summarize both 'Highly Helpful' and 'Less Helpful' reviews. These numbers help us understand the average, spread, and other important details about each group of reviews.

In simple terms, this analysis helps us see which reviews are most helpful and how their scores

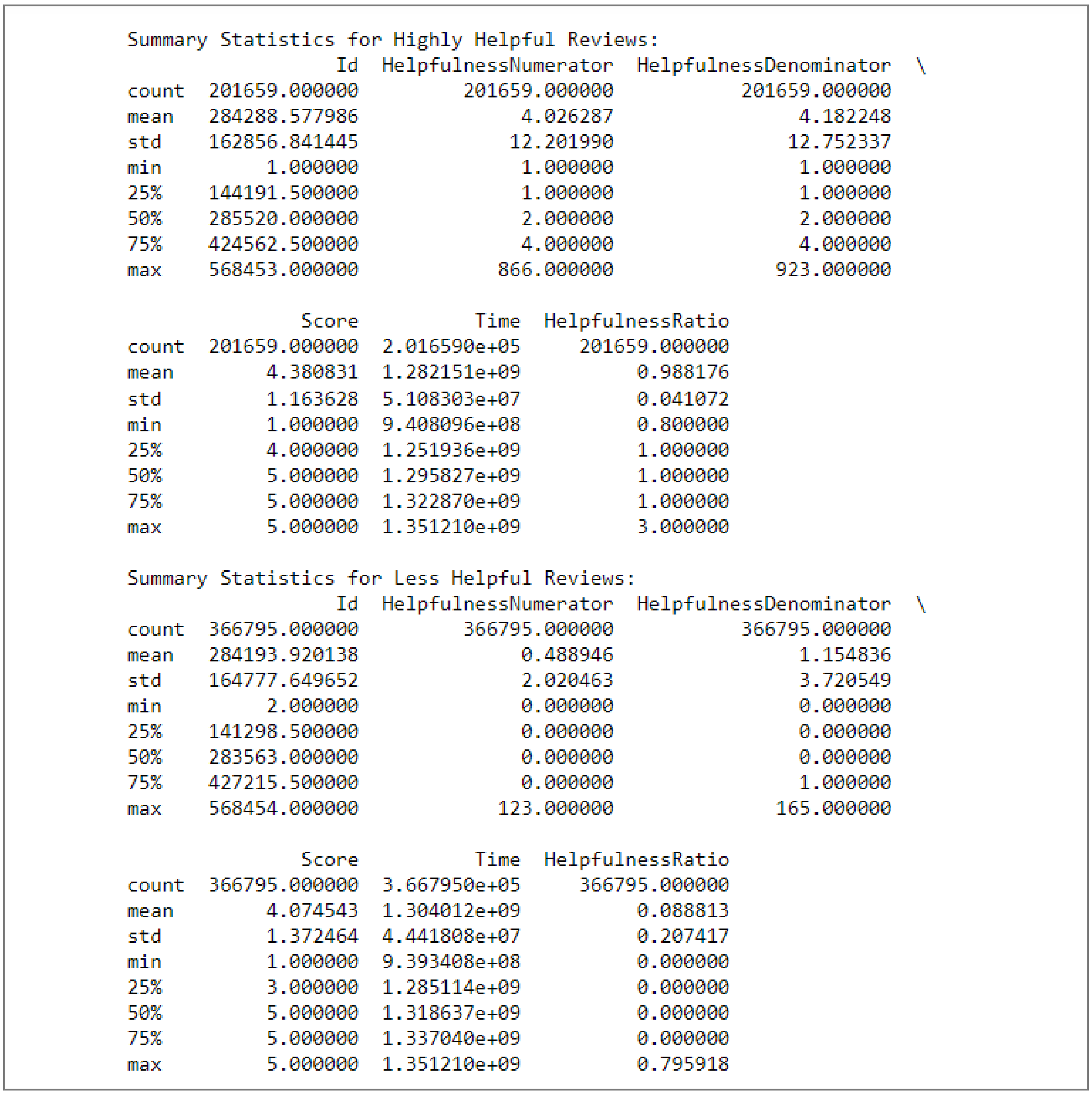


relate to their helpfulness. It gives us useful insights into user feedback and can be valuable for making informed decisions.

Fig.21. Distribution of Scores for Highly Helpful and Less Helpful Reviews

The chart provides a visual representation of how scores are distributed within these categories. Each bar in the chart corresponds to a specific score, and the height of the bars indicates the count or frequency of reviews with that score.

The chart is divided into two sections, one for reviews categorized as 'Highly Helpful' and another for reviews categorized as 'Less Helpful.' By examining the chart, one can observe how scores vary among reviews in these distinct helpfulness categories.



```
Summary Statistics for Highly Helpful Reviews:
                  Id  HelpfulnessNumerator  HelpfulnessDenominator  \
count  201659.000000         201659.000000           201659.000000
mean   284288.577986              4.026287                4.182248
std    162856.841445             12.201990               12.752337
min         1.000000              1.000000                1.000000
25%    144191.500000              1.000000                1.000000
50%    285520.000000              2.000000                2.000000
75%    424562.500000              4.000000                4.000000
max    568453.000000            866.000000              923.000000

               Score          Time  HelpfulnessRatio
count  201659.000000  2.016590e+05     201659.000000
mean        4.380831  1.282151e+09          0.988176
std         1.163628  5.108303e+07          0.041072
min         1.000000  9.408096e+08          0.800000
25%         4.000000  1.251936e+09          1.000000
50%         5.000000  1.295827e+09          1.000000
75%         5.000000  1.322870e+09          1.000000
max         5.000000  1.351210e+09          3.000000

Summary Statistics for Less Helpful Reviews:
                  Id  HelpfulnessNumerator  HelpfulnessDenominator  \
count  366795.000000         366795.000000           366795.000000
mean   284193.920138              0.488946                1.154836
std    164777.649652              2.020463                3.720549
min         2.000000              0.000000                0.000000
25%    141298.500000              0.000000                0.000000
50%    283563.000000              0.000000                0.000000
75%    427215.500000              0.000000                1.000000
max    568454.000000            123.000000              165.000000

               Score          Time  HelpfulnessRatio
count  366795.000000  3.667950e+05     366795.000000
mean        4.074543  1.304012e+09          0.088813
std         1.372464  4.441808e+07          0.207417
min         1.000000  9.393408e+08          0.000000
25%         3.000000  1.285114e+09          0.000000
50%         5.000000  1.318637e+09          0.000000
75%         5.000000  1.337040e+09          0.000000
max         5.000000  1.351210e+09          0.795918
```

Fig.22.

The summary statistics offer a detailed glimpse into two distinct categories of reviews: 'Highly Helpful' and 'Less Helpful.' For 'Highly Helpful' reviews, the average HelpfulnessNumerator stands at approximately 4, with a standard deviation of 12.20, showcasing considerable variability. The average score is 4.38, indicating predominantly positive sentiments in this category. Most of these reviews exhibit a HelpfulnessRatio of 1, denoting a high level of perceived helpfulness.

In contrast, 'Less Helpful' reviews exhibit a noticeably lower average HelpfulnessNumerator of 0.49, emphasizing a reduced degree of perceived helpfulnes.

The average score for this category is 4.07, reflecting a slightly less positive sentiment compared to 'Highly Helpful' reviews. Moreover, the HelpfulnessRatio for most reviews in this category is 0, indicating a diminished perceived helpfulness among users.

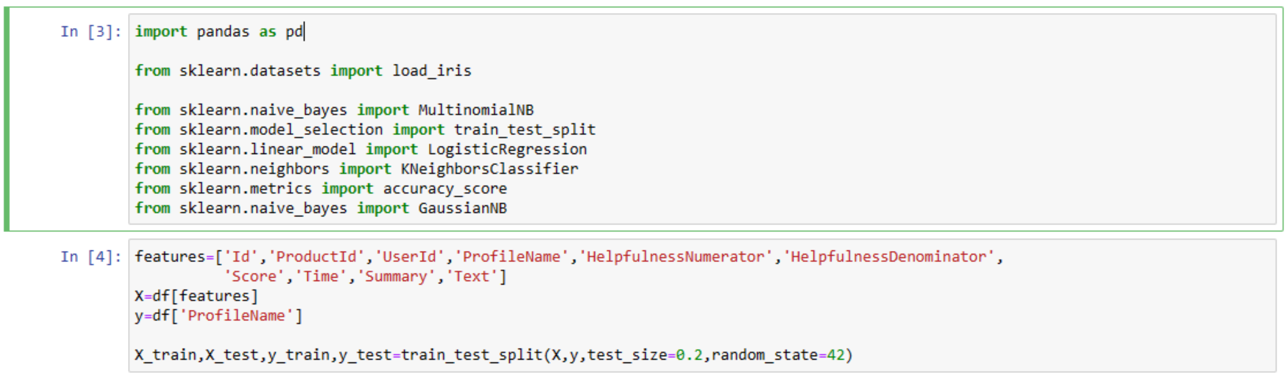
The summary statistics further reveal the spread of reviews in each category, highlighting central tendencies and variations. Overall, these statistics serve as a valuable tool for understanding the distinctive characteristics of reviews classified based on their perceived helpfulness, providing nuanced insights into user sentiments and feedback within the dataset

6. MACHINE LEARNING

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model from sample data, known as "training data", to make predictions or decisions without being explicitly programmed to perform the task.

6.1. Imports and trening data

Before I start with any application of machine learning, it is necessary to import a couple of libraries and set certain data from the data set (features), as well as trening data, with which we will perform tests and machine learning.

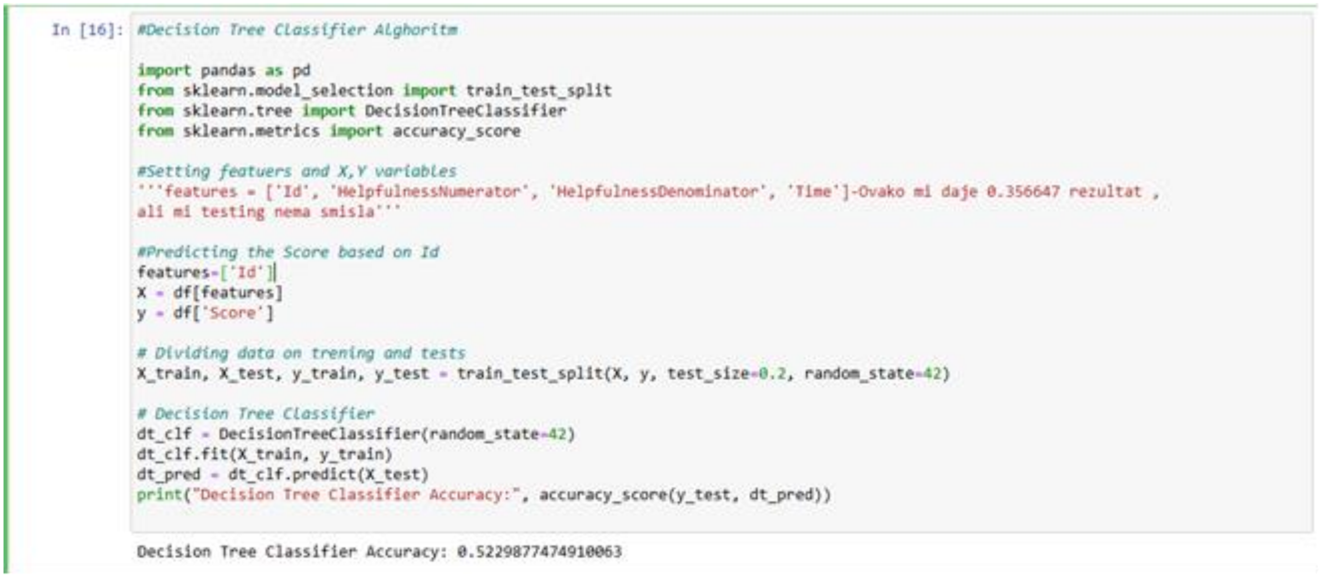


```
In [3]: import pandas as pd

        from sklearn.datasets import load_iris

        from sklearn.naive_bayes import MultinomialNB
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LogisticRegression
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.metrics import accuracy_score
        from sklearn.naive_bayes import GaussianNB
```

```
In [4]: features=['Id','ProductId','UserId','ProfileName','HelpfulnessNumerator','HelpfulnessDenominator',
                  'Score','Time','Summary','Text']
        X=df[features]
        y=df['ProfileName']

        X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42)
```

6.2. Decision Tree Classifier

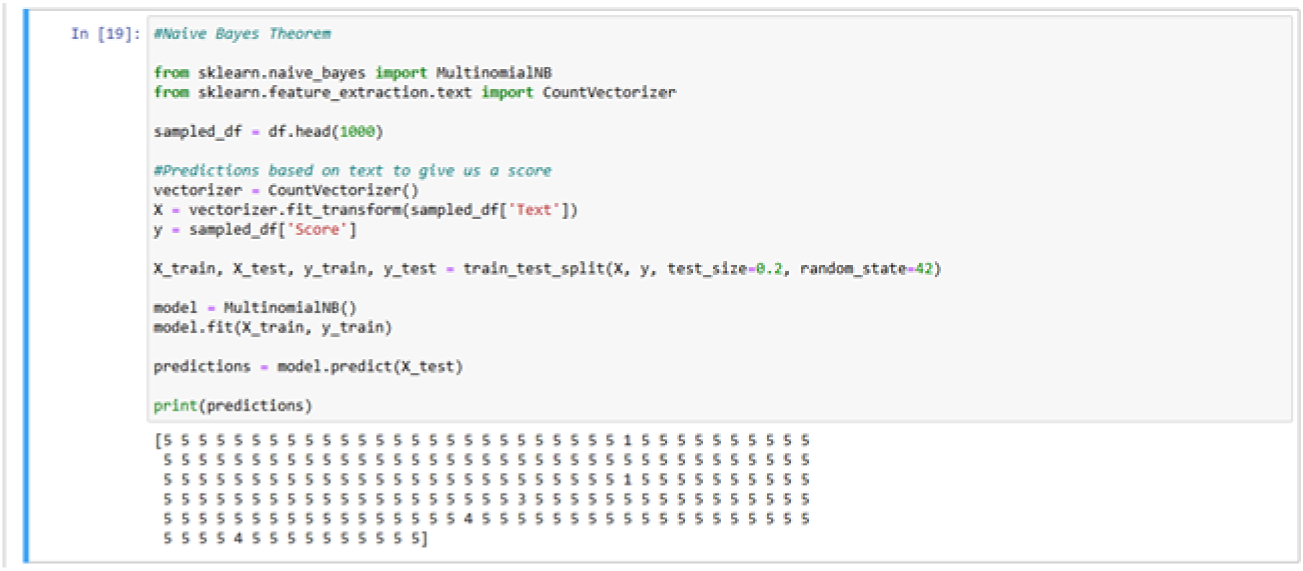
Decision Tree Classifier (DTC) is a machine learning algorithm that uses a decision tree to classify data into different classes or categories. It works by recursively partitioning the data into smaller subsets based on the values of the input features, and then making predictions based on the most common class in each subset. As X and Y parameters I use 'Id' and 'Score' columns , where I make prediction for ‘Score’ based on ‘Id’. Also, I limit my research on first 1000 rows in dataset(since the entire dataset is too long for making DTC).The result of DTC alghorithm is 0.5229877474910063 and from result of this prediction we can say that the model correctly predicted the outcome in 52.3% of cases. Since the accuracy varies between 0 and 1 (or 0% to 100%), where a higher value indicates a better performance of the model, this algorithm model with a score of 52.3% can be classified as a "solid" algorithm.



```
In [16]: #Decision Tree Classifier Alghoritm

         import pandas as pd
         from sklearn.model_selection import train_test_split
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.metrics import accuracy_score

         #Setting featuers and X,Y variables
         '''features = ['Id', 'HelpfulnessNumerator', 'HelpfulnessDenominator', 'Time']-Ovako mi daje 0.356647 rezultat ,
         ali mi testing nema smisla'''

         #Predicting the Score based on Id
         features=['Id']
         X = df[features]
         y = df['Score']

         # Dividing data on trening and tests
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # Decision Tree Classifier
         dt_clf = DecisionTreeClassifier(random_state=42)
         dt_clf.fit(X_train, y_train)
         dt_pred = dt_clf.predict(X_test)
         print("Decision Tree Classifier Accuracy:", accuracy_score(y_test, dt_pred))

         Decision Tree Classifier Accuracy: 0.5229877474910063
```

6.3. Naïve Bayes Theorem

The Naive Bayes theorem is a simplification of Bayes' theorem, a mathematical formula that describes the probability of an event given prior knowledge of the conditions that might be related to the event. In the Naive Bayes theorem, it is assumed that the features of the data are independent. After including properly libraries, I take first 1000 rows of the original dataframe and through vectorization I implement the Naive Bayes Theorem, where X and Y are my parameters 'Text' and 'Score' (respectively).With this algorithm, I wanted to make predictions based on ‘Text’ to give us a 'Score'. The results of my prediction show the following: the model made predictions of grades for the test data set. The predictions were mostly '5', with a few '1', '3' and '4' ratings. From this algorithm we can conclude that the vast majority of predictions are '5', which indicates that the model tends to predict the most common rating from the training set, but also that in addition to this rating, the algorithm also takes into account other ratings (such as 1,3,4) , although they are not as frequent.

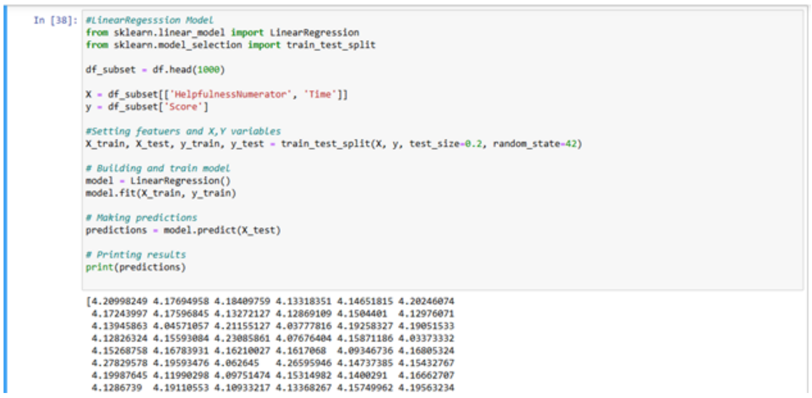


```
In [19]: #Naive Bayes Theorem

         from sklearn.naive_bayes import MultinomialNB
         from sklearn.feature_extraction.text import CountVectorizer

         sampled_df = df.head(1000)

         #Predictions based on text to give us a score
         vectorizer = CountVectorizer()
         X = vectorizer.fit_transform(sampled_df['Text'])
         y = sampled_df['Score']

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         model = MultinomialNB()
         model.fit(X_train, y_train)

         predictions = model.predict(X_test)

         print(predictions)
```

```
[5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 1 5 5 5 5 5 5 5 5 5 5
 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 1 5 5 5 5 5 5 5 5 5 5
 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 3 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
 5 5 5 5 4 5 5 5 5 5 5 5 5 5 5]
```

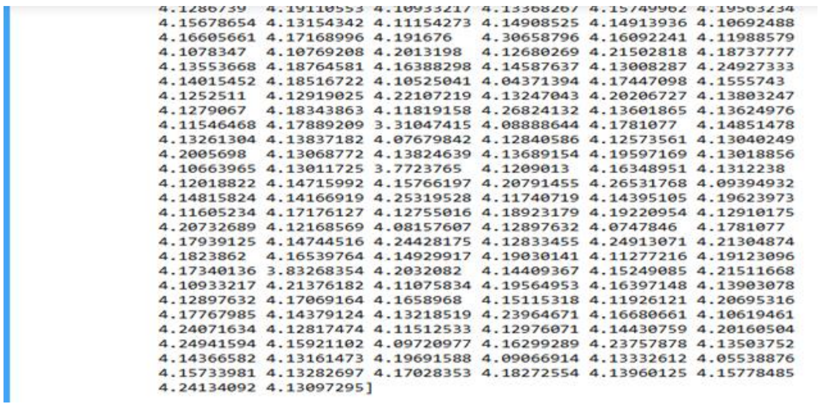
6.4. Linear regresion

Linear regression is a statistical method used to establish a linear relationship between two continuous variables. it is a way to model the relationship between a dependent variable (also called the outcome or response variable) and one or more independent variables (also called predictor or feature variables). I take first 1000 rows of dataset and for X and Y values that will go through trening process I take 'HelpfulnessNumerator', ‘Time’ and 'Score' respectively. With this algorithm, we receive predicition around the value of 4, which may mean that the average rating is in a set of data close to that value.



```
In [38]: #LinearRegesssion Model
         from sklearn.linear_model import LinearRegression
         from sklearn.model_selection import train_test_split

         df_subset = df.head(1000)

         X = df_subset[['HelpfulnessNumerator', 'Time']]
         y = df_subset['Score']

         #Setting featuers and X,Y variables
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # Building and train model
         model = LinearRegression()
         model.fit(X_train, y_train)

         # Making predictions
         predictions = model.predict(X_test)

         # Printing results
         print(predictions)
```

```
[4.20998249 4.17694958 4.18409759 4.13318351 4.14651815 4.20246074
 4.17243997 4.17596845 4.13272127 4.12869109 4.1504401  4.12976071
 4.13945863 4.04571057 4.21155127 4.03777816 4.19258327 4.19051533
 4.12826324 4.15593084 4.23085861 4.07676404 4.15871186 4.03373332
 4.15268758 4.16783931 4.16210027 4.1617068  4.09346736 4.16805324
 4.27829578 4.19593476 4.062645   4.26595946 4.14737385 4.15432767
 4.19987645 4.11990298 4.09751474 4.15314982 4.1400291  4.16662707
 4.1286739  4.19110553 4.10933217 4.13368267 4.15749962 4.19563234
```

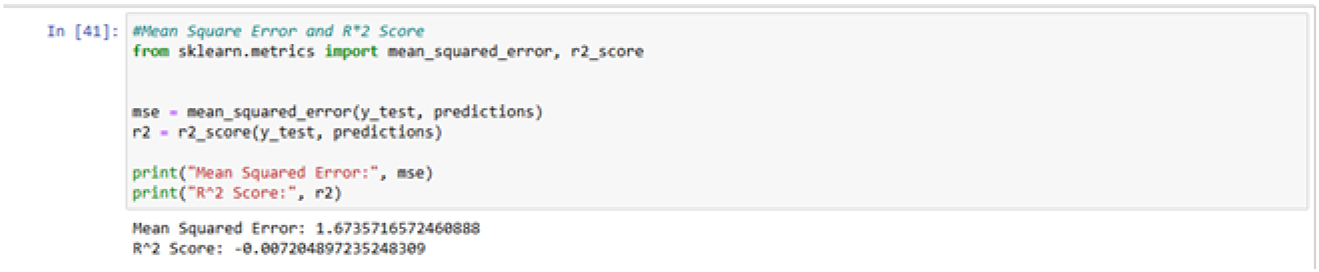


```
4.1286739  4.19110553 4.10933217 4.13368267 4.15749962 4.19563234
4.15678654 4.13154342 4.11154273 4.14908525 4.14913936 4.10692488
4.16605661 4.17168996 4.191676   4.30658796 4.16092241 4.11988579
4.1078347  4.10769208 4.2013198  4.12680269 4.21502818 4.18737777
4.13553668 4.18764581 4.16388298 4.14587637 4.13008287 4.24927333
4.14015452 4.18516722 4.10525041 4.04371394 4.17447098 4.1555743
4.1252511  4.12919025 4.22107219 4.13247043 4.20206727 4.13803247
4.1279067  4.18343863 4.11819158 4.26824132 4.13601865 4.13624976
4.11546468 4.17889209 3.31047415 4.08888644 4.1781077  4.14851478
4.13261304 4.13837182 4.07679842 4.12840586 4.12573561 4.13040249
4.2005698  4.13068772 4.13824639 4.13689154 4.19597169 4.13018856
4.10663965 4.13011725 3.7723765  4.1209013  4.16348951 4.1312238
4.12018822 4.14715992 4.15766197 4.20791455 4.26531768 4.09394932
4.14815824 4.14166919 4.25319528 4.11740719 4.14395105 4.19623973
4.11605234 4.17176127 4.12755016 4.18923179 4.19220954 4.12910175
4.20732689 4.12168569 4.08157607 4.12897632 4.0747846  4.1781077
4.17939125 4.14744516 4.24428175 4.12833455 4.24913071 4.21304874
4.1823862  4.16539764 4.14929917 4.19030141 4.11277216 4.19123096
4.17340136 3.83268354 4.2032082  4.14409367 4.15249085 4.21511668
4.10933217 4.21376182 4.11075834 4.19564953 4.16397148 4.13903078
4.12897632 4.17069164 4.1658968  4.15115318 4.11926121 4.20695316
4.17767985 4.14379124 4.13218519 4.23964671 4.16680661 4.10619461
4.24071634 4.12817474 4.11512533 4.12976071 4.14430759 4.20160504
4.24941594 4.15921102 4.09720977 4.16299289 4.23757878 4.13503752
4.14366582 4.13161473 4.19691588 4.09066914 4.13332612 4.05538876
4.15733981 4.13282697 4.17028353 4.18272554 4.13960125 4.15778485
4.24134092 4.13097295]
```

6.5. Mean square error and R*2 score

Mean Square Error and R*2 Score is alghorithm that show that the model has a certain(or low) capacity to predict ratings based on the number of useful votes and the time of publication time. MSE measures an average square error between actual and foreseen values. Lower value means a smaller mistake and a better performance of the model. R^2 Score measures how well the models of the model explain the variability of real values. The value of the closer 1 indicates a better model, while the negative value means a worse model. In my code , Mean Squared Error: 1.6735716572460888

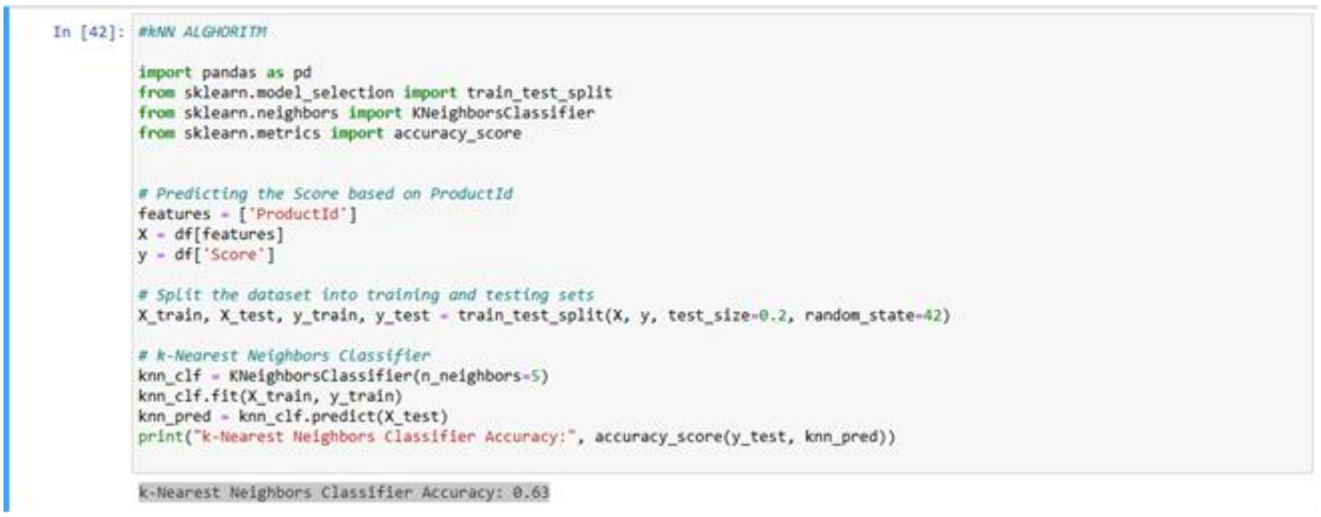
R^2 Score: -0.007204897235248309.This mean that this model is „the worst model"(for now) to deal with in my dataset.



```
In [41]: #Mean Square Error and R*2 Score
         from sklearn.metrics import mean_squared_error, r2_score


         mse = mean_squared_error(y_test, predictions)
         r2 = r2_score(y_test, predictions)

         print("Mean Squared Error:", mse)
         print("R^2 Score:", r2)

         Mean Squared Error: 1.6735716572460888
         R^2 Score: -0.007204897235248309
```

6.6. kNN algorithm

The k-Nearest Neighbors (k-NN) algorithm is a type of supervised learning algorithm that is used for classification and regression tasks. It is a simple, non-parametric, and widely used machine learning technique.In this algorithm I made prediction of 'Score' based on 'ProductId'.

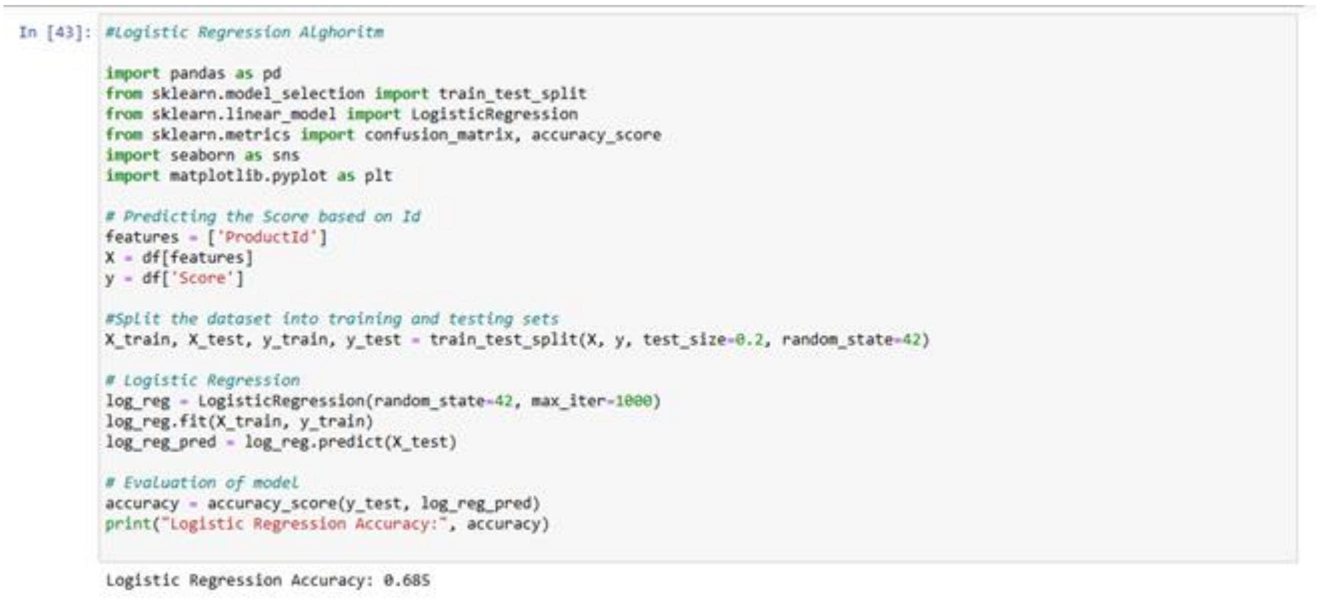


```
In [42]: #kNN ALGHORITM

         import pandas as pd
         from sklearn.model_selection import train_test_split
         from sklearn.neighbors import KNeighborsClassifier
         from sklearn.metrics import accuracy_score


         # Predicting the Score based on ProductId
         features = ['ProductId']
         X = df[features]
         y = df['Score']

         # Split the dataset into training and testing sets
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # k-Nearest Neighbors Classifier
         knn_clf = KNeighborsClassifier(n_neighbors=5)
         knn_clf.fit(X_train, y_train)
         knn_pred = knn_clf.predict(X_test)
         print("k-Nearest Neighbors Classifier Accuracy:", accuracy_score(y_test, knn_pred))

         k-Nearest Neighbors Classifier Accuracy: 0.63
```

Based on output in screenshot above , and result of 0.63 (63%) , we can place this algorithm in ‘solid’ category. The accurancy of 63% menas that the classifier works moderate.

6.7. Logistic regression algorithm

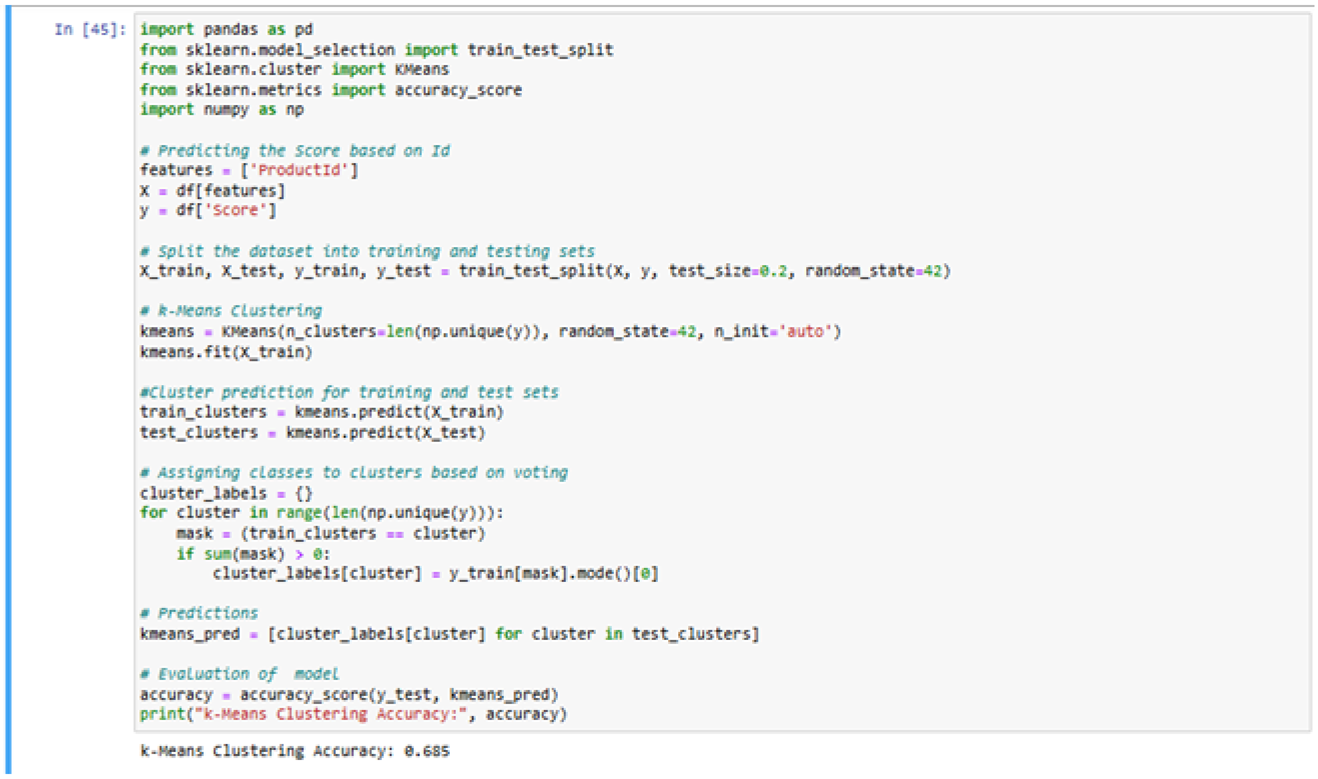
Logistic Regression is a classification algorithm used to predict the probability that a selected set of features will result in a certain classified result.I perform this alghoritm on ‘ProductId’ and ‘Score’ to make prediction of ‘Score’ based on ‘ProductId’. The result of 0.685 or (68.5%) means that this algorithm perform very well , but it’s still not on the level to be considered as ‘The best ’ for this category.



```
In [43]: #Logistic Regression Alghoritm

         import pandas as pd
         from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LogisticRegression
         from sklearn.metrics import confusion_matrix, accuracy_score
         import seaborn as sns
         import matplotlib.pyplot as plt

         # Predicting the Score based on Id
         features = ['ProductId']
         X = df[features]
         y = df['Score']

         #Split the dataset into training and testing sets
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # Logistic Regression
         log_reg = LogisticRegression(random_state=42, max_iter=1000)
         log_reg.fit(X_train, y_train)
         log_reg_pred = log_reg.predict(X_test)

         # Evaluation of model
         accuracy = accuracy_score(y_test, log_reg_pred)
         print("Logistic Regression Accuracy:", accuracy)

         Logistic Regression Accuracy: 0.685
```

6.8. kMeans algorithm

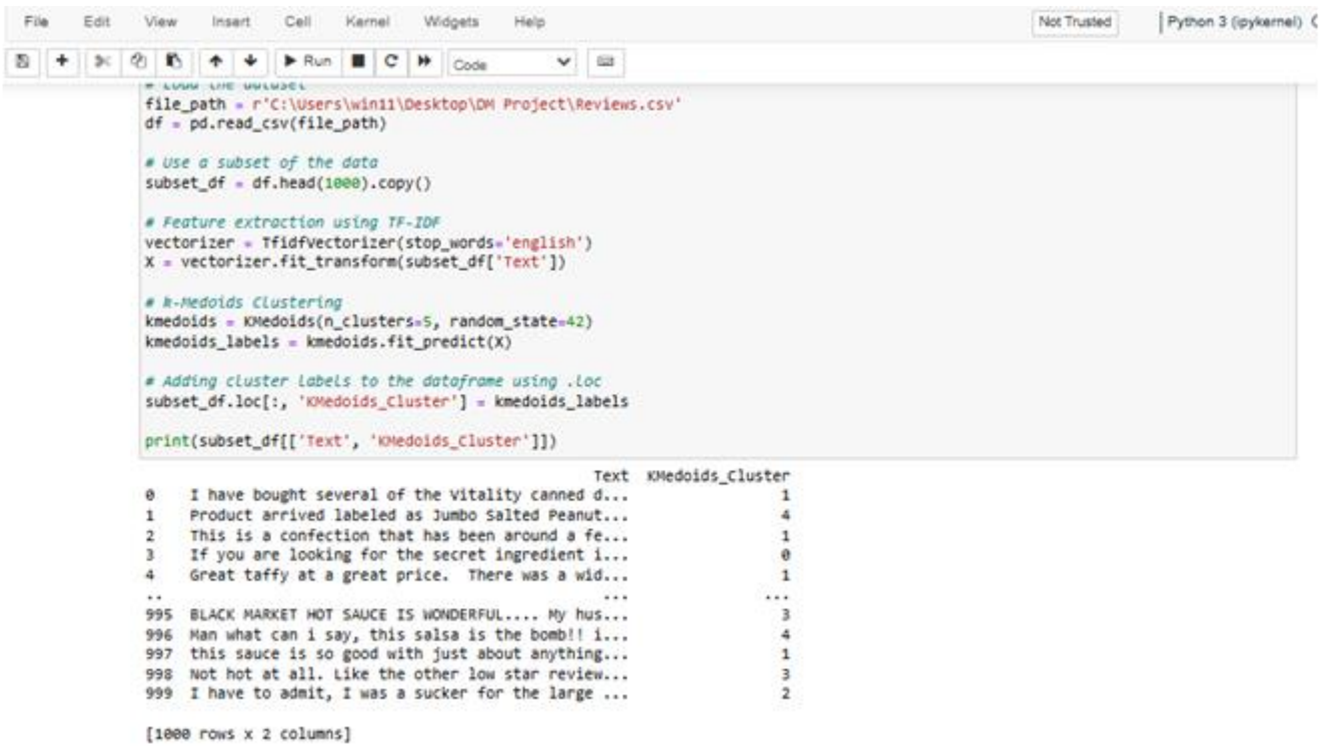
K-Means is a type of unsupervised machine learning algorithm that is used to cluster data into K groups, or clusters, based on their similarities. This algorithm uses k-Means Clustering to group data into clusters (groups). Clusters are closed groups that have similar characteristics (signs). In my case, the number of clusters is equal to the number of unique values in the column 'Score' (n_clusters=len(np.unique(y))). The result of this algorithm based on the column "ProductId" and further prediction of the value of the column "Score", tells us that the algorithm is successful in about 68.5% of cases.



```
In [45]: import pandas as pd
         from sklearn.model_selection import train_test_split
         from sklearn.cluster import KMeans
         from sklearn.metrics import accuracy_score
         import numpy as np

         # Predicting the Score based on Id
         features = ['ProductId']
         X = df[features]
         y = df['Score']

         # Split the dataset into training and testing sets
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # k-Means Clustering
         kmeans = KMeans(n_clusters=len(np.unique(y)), random_state=42, n_init='auto')
         kmeans.fit(X_train)

         #Cluster prediction for training and test sets
         train_clusters = kmeans.predict(X_train)
         test_clusters = kmeans.predict(X_test)

         # Assigning classes to clusters based on voting
         cluster_labels = {}
         for cluster in range(len(np.unique(y))):
             mask = (train_clusters == cluster)
             if sum(mask) > 0:
                 cluster_labels[cluster] = y_train[mask].mode()[0]

         # Predictions
         kmeans_pred = [cluster_labels[cluster] for cluster in test_clusters]

         # Evaluation of  model
         accuracy = accuracy_score(y_test, kmeans_pred)
         print("k-Means Clustering Accuracy:", accuracy)

         k-Means Clustering Accuracy: 0.685
```

6.9. the DBSCAN(Density-Based Spatial Clustering of Applications with Noise) algorithm

The DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm is used to detect groups of elements that are close and similar, and to identify anomalies (noise) in the data.From output above (based on first 1000 rows) we can say that The cluster labels are integers, with -1 indicating that the sample was not part of a dense region (i.e., it was not clustered) and positive integers indicating the cluster number.In this case, all 5 samples were assigned a label of -1, which means that they did not form a dense region and were not clustered together.

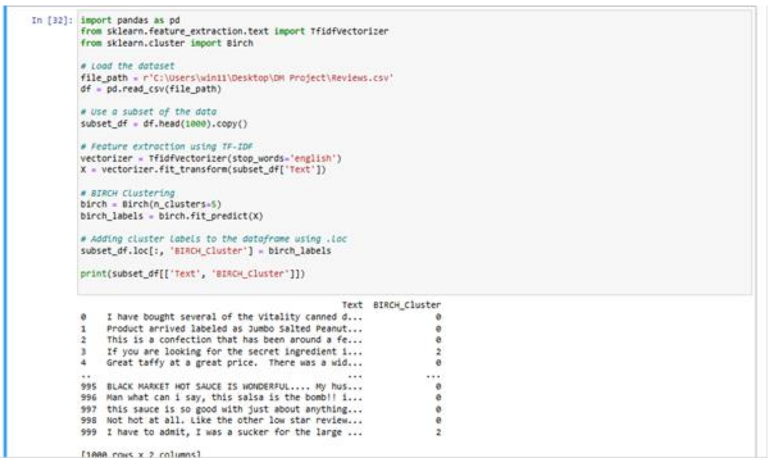


```
# Load the dataset
file_path = r'C:\Users\win11\Desktop\DM Project\Reviews.csv'
df = pd.read_csv(file_path)

# Use a subset of the data
subset_df = df.head(1000).copy()

# Feature extraction using TF-IDF
vectorizer = TfidfVectorizer(stop_words='english')
X = vectorizer.fit_transform(subset_df['Text'])

# K-Medoids Clustering
kmedoids = KMedoids(n_clusters=5, random_state=42)
kmedoids_labels = kmedoids.fit_predict(X)

# Adding cluster labels to the dataframe using .loc
subset_df.loc[:, 'KMedoids_Cluster'] = kmedoids_labels

print(subset_df[['Text', 'KMedoids_Cluster']])
```

```
                                                  Text  KMedoids_Cluster
0    I have bought several of the Vitality canned d...                 1
1    Product arrived labeled as Jumbo Salted Peanut...                 4
2    This is a confection that has been around a fe...                 1
3    If you are looking for the secret ingredient i...                 0
4    Great taffy at a great price.  There was a wid...                 1
..                                                 ...               ...
995  BLACK MARKET HOT SAUCE IS WONDERFUL.... My hus...                 3
996  Man what can i say, this salsa is the bomb!! i...                 4
997  this sauce is so good with just about anything...                 1
998  Not hot at all. Like the other low star review...                 3
999  I have to admit, I was a sucker for the large ...                 2

[1000 rows x 2 columns]
```

6.10. BIRCH cluster algorithm

In this algorithm, I wanted to classify the entire text from column 'Text' into specific clusters. From the output below, we can conclude that 5 clusters have been created, and we can see which texts belong to which cluster. For example, the first few reviews belong to cluster 1, and the rest belong to different clusters. This means that those reviews are similar to each other, and the differences are between clusters.

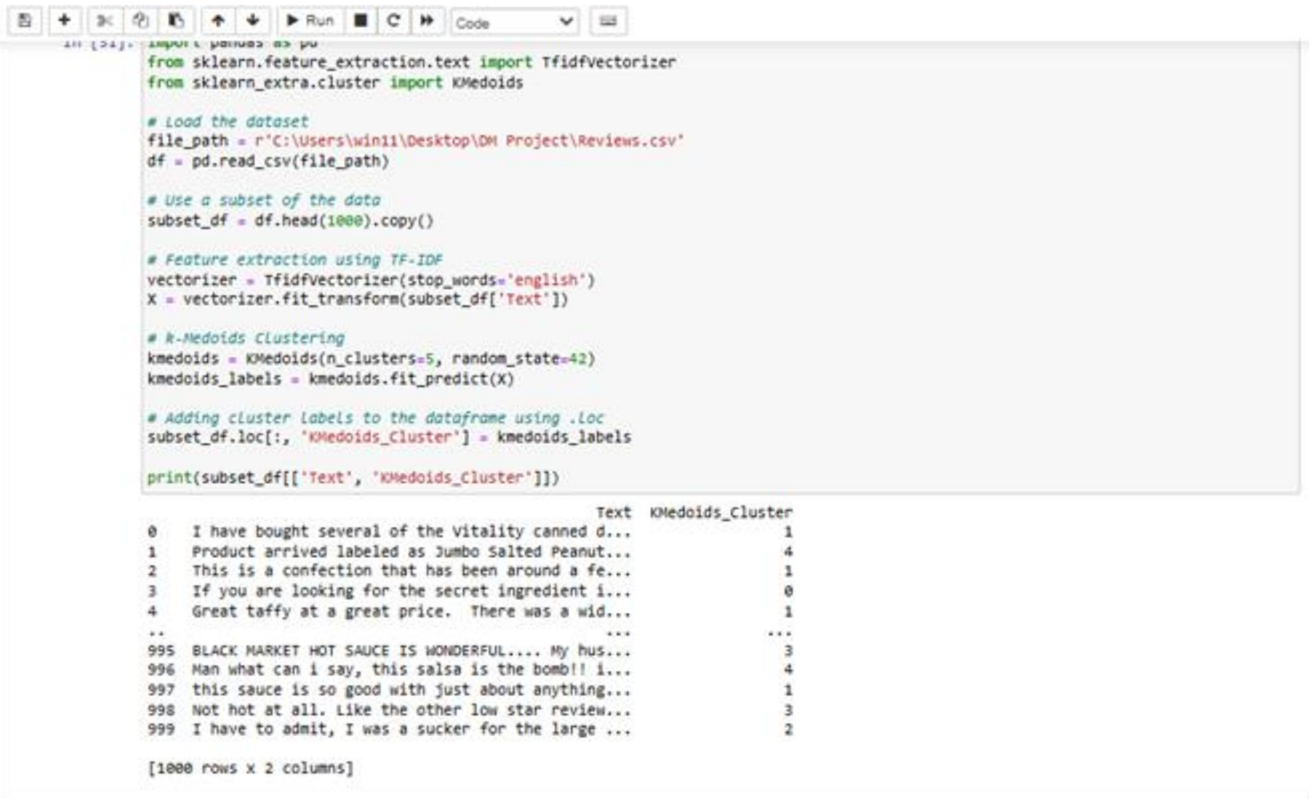


```
In [32]: import pandas as pd
         from sklearn.feature_extraction.text import TfidfVectorizer
         from sklearn.cluster import Birch

         # Load the dataset
         file_path = r'C:\Users\win11\Desktop\DM Project\Reviews.csv'
         df = pd.read_csv(file_path)

         # Use a subset of the data
         subset_df = df.head(1000).copy()

         # Feature extraction using TF-IDF
         vectorizer = TfidfVectorizer(stop_words='english')
         X = vectorizer.fit_transform(subset_df['Text'])

         # BIRCH Clustering
         birch = Birch(n_clusters=5)
         birch_labels = birch.fit_predict(X)

         # Adding cluster labels to the dataframe using .loc
         subset_df.loc[:, 'BIRCH_Cluster'] = birch_labels

         print(subset_df[['Text', 'BIRCH_Cluster']])
```

```
                                                  Text  BIRCH_Cluster
0    I have bought several of the Vitality canned d...              0
1    Product arrived labeled as Jumbo Salted Peanut...              0
2    This is a confection that has been around a fe...              0
3    If you are looking for the secret ingredient i...              2
4    Great taffy at a great price.  There was a wid...              0
..                                                 ...            ...
995  BLACK MARKET HOT SAUCE IS WONDERFUL.... My hus...              0
996  Man what can i say, this salsa is the bomb!! i...              0
997  this sauce is so good with just about anything...              0
998  Not hot at all. Like the other low star review...              0
999  I have to admit, I was a sucker for the large ...              2

[1000 rows x 2 columns]
```

6.11. Kmedodis algorithm

KMedoids is a clustering algorithm that uses the medoids (objects that are most representative of their clusters) to group similar objects together.Medoids are objects that are chosen as representatives of their clusters. In KMedoids, the medoids are selected based on the minimum total dissimilarity within each cluster. In this algorithm, I wanted to classify the entire text (but only first 1000 rows) from column 'Text' into specific clusters. From the output below, we can conclude that 5 clusters have been created, and we can see which texts belong to which cluster.



```
In [31]: import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn_extra.cluster import KMedoids

# Load the dataset
file_path = r'C:\Users\win11\Desktop\DM Project\Reviews.csv'
df = pd.read_csv(file_path)

# Use a subset of the data
subset_df = df.head(1000).copy()

# Feature extraction using TF-IDF
vectorizer = TfidfVectorizer(stop_words='english')
X = vectorizer.fit_transform(subset_df['Text'])

# k-Medoids Clustering
kmedoids = KMedoids(n_clusters=5, random_state=42)
kmedoids_labels = kmedoids.fit_predict(X)

# Adding cluster labels to the dataframe using .loc
subset_df.loc[:, 'KMedoids_Cluster'] = kmedoids_labels

print(subset_df[['Text', 'KMedoids_Cluster']])
```

```
                                                  Text  KMedoids_Cluster
0    I have bought several of the Vitality canned d...                 1
1    Product arrived labeled as Jumbo Salted Peanut...                 4
2    This is a confection that has been around a fe...                 1
3    If you are looking for the secret ingredient i...                 0
4    Great taffy at a great price.  There was a wid...                 1
..                                                 ...               ...
995  BLACK MARKET HOT SAUCE IS WONDERFUL.... My hus...                 3
996  Man what can i say, this salsa is the bomb!! i...                 4
997  this sauce is so good with just about anything...                 1
998  Not hot at all. Like the other low star review...                 3
999  I have to admit, I was a sucker for the large ...                 2

[1000 rows x 2 columns]
```

6.12. Random forest algorithm

Random Forest is an ensemble learning method that combines multiple decision trees to improve the accuracy and robustness of predictions.In my code , this alghoritm give me a result of 0.81 or 81% which is the faster and better in all others alghoritms. Again, I use ‘ProductId’ and ‘Score’ in this algorithm to predict Score based on ProductId.

6.13. AdaBoost algorithm

AdaBoost (short for Adaptive Boosting) is a popular machine learning algorithm for boosting the performance of weak classifiers. It is a type of ensemble learning method that combines multiple weak classifiers to create a strong classifier.In my code , this alghoritm give me a result of 0.87 or 87% which is the faster and better in all others alghoritms. Again, I use ‘ProductId’ and ‘Score’ in this algorithm to predict Score based on ProductId.

CONCLUSION

In conclusion, this comprehensive analysis of the fine food reviews dataset from Amazon provides valuable insights into user sentiments, preferences, and linguistic patterns. The dataset, spanning from October 1999 to October 2012, encompasses a vast collection of 568,454 reviews from 256,059 unique users on 74,258 products.

The project employed a robust methodology, starting with data loading, cleaning, and text preprocessing, ensuring the integrity of the dataset. The subsequent analyses covered diverse aspects, including word frequency, temporal trends, product reviews, review length, score distribution, user engagement, slang words, word cloud generation, top reviewers over time, and helpfulness analysis.

Key findings include the identification of highly-reviewed products, understanding temporal shifts in review activity, recognizing influential users, and unveiling prevalent slang words. The word cloud visually highlighted frequently used terms, providing an intuitive representation of dominant themes. The analysis of top reviewers over time showcased the consistent contributors to the dataset.

The helpfulness analysis offered valuable insights into the relationship between review scores and perceived helpfulness. This information is crucial for businesses and platforms to understand user feedback dynamics and make informed decisions for product improvements or marketing strategies.

In this project, I also explored the application of machine learning algorithms to analyze and predict the relationships between various features in a dataset.

The use of algorithms like: kMeans, KMedoids, BIRCH Cluster Algorithm, The DBSCAN, 7.7 Logistic Regression Alghorithm, Linear regression, Naive Bayes Theorem, Descion tree, kNN, ADABOOST, RandomForest algorithm in this dataset,

I additionally showed correlations between related columns in the dataset , their similarity and relationship through numerical parameters (numbers). Also, these algorithms helped me to prove that their use greatly helps in classifying a huge amount of text within large datasets (like mine), and that they can be grouped based on similarities or differences such as: reviews, IDs, etc. By using all of the above as well as algorithms, a more detailed insight into the dataset of a huge "story" (Amazon Fine Food Reviews) is given, and through the presented analysis, a better explanation and overview of the reviews is available to the user, which largely determines the fact whether the person who reads be the next user/customer.

REFERENCES

1. What is Data Mining ?(26.11.2023)
2. What is Exploratory Data Analysis? | IBM (26.11.2023)
3. Intro-to-exploratory-data-analysis-eda-in-python (03.12.2023)
4. Hands-On Exploratory Data Analysis with Python (07.12.2023)
5. Cleaning dataset in Python [07.12.2023]
6. Exploratory data analysis on Netflix Dataset (15.12.2023)
7. ZOMATO DATA WITH EDA, GEOSPATIAL AND SENTIMENT ANALYSIS (21.12.2023)
8. Exploratory Data Analysis and Latent Dirichlet Allocation (21.12.2023)
9. eda-exploratory-data-analysis-project-using-python-de90cbf4e128 (16.12.2023)
10. Step-by-Step Exploratory Data Analysis (EDA) using Python (21.12.2023)
11. https://www.geeksforgeeks.org/what-is-exploratory-data-analysis/ (16.12.2023)
12. simplilearn.com/tutorials/data-analytics-tutorial/exploratory-data-analysis (17.12.2023)
13. EDA (18.12.2023)
14. exploratory-data-analysis-and-visualization-techniques-in-data-science/ (18.12.2023)
15. /exploratory-data-analysis/ (19.12.2023) [16] What is EDA? (19.12.2023)
16. exploratory-data-analysis-step-by-step-guide-for-data-analyst (20.12.2023)
17. Why is exploratory data analysis important? (20.12.2023)
18. data-science/eda-data-science (20.12.2023)
19. Cleaning data (20.12.2023)
20. Naive Bayes (2024)
21. Machine Learning(2024)
22. What is Decistion Tree Classificator (DTC) ?