INTERNATIONAL BURCH UNIVERSITY

FACULTY OF ENGINEERING AND NATURAL SCIENCES

DEPARTMENT OF INFORMATION TECHNOLOGIES

Helpfulness Analysis & Prediction of Amazon Food Reviews

PROJECT PAPER

EMAN HRUSTEMOVIĆ

Supervisor:

Assoc. Prof. Dr. Zerina Altoka

SARAJEVO

January, 2026

TABLE OF CONTENTS

ABSTRACT

The objective of this study is to analyze and predict the helpfulness of online product reviews in order to better understand the factors that influence how users perceive and engage with customer feedback on e-commerce platforms. By examining the relationship between review characteristics and perceived helpfulness, this research aims to provide actionable insights that can improve user experience and enhance the overall quality of review systems.

The research employs exploratory data analysis (EDA) and statistical techniques to identify trends, distributions, and relationships among key variables, including review length, rating score, and temporal attributes. The dataset, consisting of Amazon food product reviews, is processed to compute the helpfulness ratio based on user voting behavior. Temporal patterns, numerical correlations, and inferential statistical tests—such as correlation analysis, independent t-tests, and one-way ANOVA—are applied to examine the impact of review features on helpfulness.

In addition, a simple predictive model based on linear regression is developed to quantify the influence of selected review attributes on the helpfulness ratio. The model emphasizes interpretability rather than predictive complexity, allowing for clear insights into the direction and magnitude of feature effects.

All data processing, analysis, and visualization tasks are conducted using Python, with libraries such as pandas, seaborn, and scikit-learn. Reproducibility is ensured through modular code design and controlled random seed initialization. The results indicate that review length, rating score, and temporal factors play a significant role in determining perceived review helpfulness.

This study demonstrates the value of data science methodologies in analyzing user-generated content and improving the design of review and recommendation systems. Future work may extend this analysis by incorporating natural language processing techniques to examine textual sentiment and semantic features, as well as by expanding the dataset to include multiple product categories and platforms.

Keywords: Data Science, Helpfulness Ratio, Review Analysis, Exploratory Data Analysis, Linear Regression, Statistical Inference, Amazon Food Reviews, User Engagement

1. INTRODUCTION

In the contemporary digital landscape, online reviews have become a critical source of information for consumers worldwide. They provide insights into product quality, user experience, and overall satisfaction, significantly influencing purchasing decisions and shaping public opinion. E-commerce platforms such as Amazon allow customers to share their experiences and opinions, resulting in large volumes of user-generated content across a wide range of products (Mudambi & Schuff, 2010). For consumers, reviews function as decision-support tools, while for sellers, they represent valuable feedback for improving product offerings and services(Pan & Zhang, 2011).

As the volume of online reviews continues to grow, extracting meaningful and reliable insights from this information becomes increasingly challenging. Potential buyers are often confronted with contradictory opinions, vague statements, and an overwhelming number of reviews. In this context, identifying which reviews are genuinely helpful is essential for improving the usability and effectiveness of review systems. The concept of review helpfulness—typically determined through user voting mechanisms—serves as a practical metric for highlighting reviews that provide the most value to readers.

This study focuses on analyzing the helpfulness of Amazon food product reviews using data science methodologies. Leveraging a large-scale dataset spanning more than a decade and comprising approximately 568,000 reviews, the research examines how various review characteristics—such as word count, rating score, review length, and temporal attributes—relate to perceived helpfulness. Exploratory Data Analysis (EDA) and statistical techniques are applied to uncover patterns, distributions, and correlations within the data, offering a deeper understanding of user engagement with reviews.

A central component of this research is the calculation of the helpfulness ratio, which quantifies the proportion of positive helpfulness votes relative to total votes received by a review. Temporal analysis is employed to identify long-term and seasonal trends in review behavior, while inferential statistical tests—including independent t-tests and one-way ANOVA—are used to assess significant differences between groups of reviews. In addition, a simple linear regression model is developed to quantify the influence of selected review features, such as word count, rating score, and year, on the helpfulness ratio.

The primary objective of this research is to provide actionable insights into the factors that drive review helpfulness on e-commerce platforms. By emphasizing interpretability, statistical rigor, and reproducibility, this study demonstrates how data science approaches can be used to analyze user-generated content and improve the design of review and recommendation systems. Ultimately, the findings aim to benefit both consumers—by helping them identify informative reviews—and sellers—by offering a clearer understanding of what constitutes helpful customer feedback.

2. Literature Review

The analysis of online reviews has become a prominent area of research within e-commerce analytics, driven by the increasing reliance of consumers on user-generated content for informed decision-making. Large-scale platforms such as Amazon provide vast amounts of review data, offering valuable opportunities to study customer behavior, engagement patterns, and factors that influence perceived review usefulness. Prior research has emphasized statistical and exploratory approaches to examine review characteristics, with particular attention given to the concept of review helpfulness as a measurable outcome of user interaction.

Review helpfulness, typically measured through user voting mechanisms, has been widely studied as a proxy for perceived review quality. Mudambi and Schuff (2010) were among the first to empirically demonstrate that review depth and extremity significantly influence helpfulness ratings, highlighting the importance of textual and numerical features. Their findings established a foundation for subsequent research focusing on the structural properties of reviews rather than solely on sentiment.Review helpfulness is commonly used as an indicator of review quality and user-perceived usefulness on e-commerce platforms (Mudambi & Schuff, 2010; Filieri, 2015).

Temporal aspects of review helpfulness have also received increasing attention in the literature. Chen, Dhanasobhon, and Smith (2008) demonstrated that review timing and exposure duration significantly affect helpfulness votes, emphasizing the importance of time-based analysis in review systems. More recent studies further confirmed that user engagement with reviews evolves over time, with older reviews often accumulating higher helpfulness scores due to increased visibility and credibility.

Classical statistical methods play a central role in understanding relationships between review attributes and perceived helpfulness. Independent sample t-tests and analysis of variance (ANOVA) have been used to compare helpfulness across rating groups, while correlation analyses have been employed to quantify associations between review length, rating scores, and helpfulness ratios.Classical statistical methods, such as the independent t-test and ANOVA, have been widely adopted in review analytics (Chen et al., 2008; Mudambi & Schuff, 2010). For instance, Ghose and Ipeirotis (2011) demonstrated that star ratings and review length jointly influence perceived review usefulness, reinforcing the relevance of descriptive and inferential statistics in review analytics.

Regression-based approaches have further enabled researchers to quantify the impact of multiple review features on helpfulness outcomes. Studies using linear and logistic regression models have shown that longer reviews with moderate sentiment polarity tend to receive higher helpfulness scores, although the magnitude of these effects remains relatively modest. Such findings underscore the importance of interpretability-focused models when analyzing user-generated content, particularly in exploratory data science contexts.

Existing literature consistently highlights the value of user-provided helpfulness votes—often operationalized as a ratio of positive votes to total votes—as a robust indicator of review usefulness. By modeling helpfulness as a continuous variable, researchers can capture nuanced differences in user perception while maintaining statistical rigor. Moreover, time-based analyses suggest that helpfulness is influenced not only by review content but also by broader temporal and behavioral trends within online platforms.Several studies suggest that review depth and rating extremity influence perceived helpfulness, although their effects vary across datasets (Mudambi & Schuff, 2010; Ghose & Ipeirotis, 2011).

Overall, prior research supports the use of exploratory data analysis, statistical inference, and simple predictive modeling to investigate review helpfulness. Building upon these established methodologies, the present study adopts a data science–oriented approach to analyze Amazon food reviews, emphasizing interpretability, reproducibility, and empirical validation. Regression-based approaches have been widely applied to model review helpfulness using structured numerical features (Liu et al., 2008; Ghose & Ipeirotis, 2011).

Table 2.1: Tabular analysis of reviewed literature

Year	Author	Algorithms / Methods	Dataset	Key Finding	References (with DOI)
2007	Liu, J., Cao, Y., Lin, C.-Y., Huang, Y., & Zhou, M.	Feature engineering, text analysis	Online product reviews	Low-quality review detection; improves summarization	https://aclanthology.org/D07-1035/
2008	Forman, Ghose & Wiesenfeld	Statistical modeling	Amazon reviews	Reviewer identity changes perceived usefulness	https://pubsonline.informs.org/doi/10.1287/isre.1080.0193
2008	Chen, P.-Y., Dhanasobhon, S., & Smith, M. D.	Statistical analysis, disaggregate regression	Amazon.com reviews	Reviewer & content factors affect helpfulness and impact on sales	https://doi.org/10.2139/ssrn.918083
2010	Mudambi & Schuff	Regression, descriptive analysis	Amazon reviews	Depth & extremity	https://doi.org/10.2307/20721420

				influence helpfulness	
2011	Ghose, A.; Ipeirotis, P. G.	Linear Regression modeling	Amazon reviews	Star ratings & length predict helpfulness	https://doi.org/10.1109/TKDE.2010.188

3. Materials and Methodology

3.1 Dataset

The dataset used in this research is “Amazon Fine Food Reviews” dataset from Kaggle.It contains detailed information about food product reviews collected from the Amazon platform. The dataset is suitable for exploratory data analysis and statistical evaluation of user-generated reviews, enabling the analysis of review characteristics and their relationship with perceived helpfulness. It is provided with multiple features that can help identify product ratings, user information, and textual content of the reviews.The dataset contains reviews of fine foods from amazon. The data span a period of more than 10 years, including all ~500,000 reviews up to October 2012. Reviews include product and user information, ratings, and a plain text review.The dataset focuses exclusively on fine food products reviewed on the Amazon platform.The dataset contains :

1. Reviews from Oct 1999 - Oct 2012
2. 568,454 reviews
3. 256,059 users
4. 74,258 products
5. 260 users with > 50 reviews

Below is a detailed explanation of each column in the dataset.

Table 3.1: Analysis of the Product Reviews Amazon Fine Food Dataset

Column Name	**Information**
Id	Unique identifier for each review record
ProductId	Unique identifier for every product
UserId	Unique identifier for every user
ProfileName	Profile name of user who posted review
HelpfulnessNumerator	The numerator value of helpfulness for the review, indicating how helpful users found it.
HelpfulnessDenominator	The denominator value of helpfulness for the review, indicating the total number of users who found it helpful.
Score	The product rating, typically ranging from 1 to 5 stars.
Time	Timestamp of when the review was posted.
Summary	A brief summary of the review content.

Text	The full text of the review, providing detailed feedback and comments from the user.

The structure of the dataset indicates that several preprocessing steps need to be taken to ensure data quality and reliability for exploratory data analysis, statistical modeling, and visualization. Clean data is fundamental to deriving meaningful insights, and any inconsistencies, missing values, or redundant entries in the dataset require careful handling.

One of the primary challenges is dealing with missing or null values in certain columns. While most columns in the dataset have no missing data, exceptions exist wherein a small number of records have null values. Identifying and addressing these missing values is important to ensure reliable statistical outputs and minimize bias during the analysis phase.The number of null values in each column was checked, and the results are summarized in following table:

Column Name	**Number of Null Values**
Id	0
ProductId	0
UserId	0
ProfileName	26
HelpfulnessNumerator	0
HelpfulnessDenominator	0
Score	0
Time	0
Summary	27

Text	0

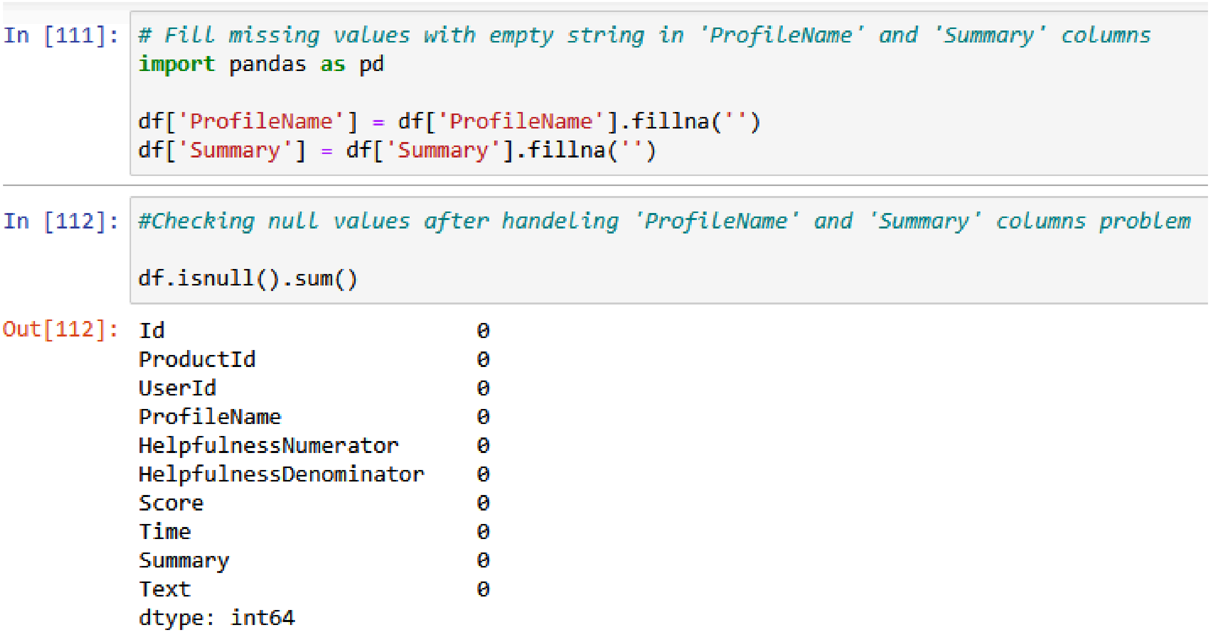
From the table above, it can be seen that the dataset does not have a significant number of missing values. Specifically, the ProfileName column has 26 missing values, and the Summary column has 27 missing values. Considering the size of the dataset, with over 568,000 reviews, these missing values represent an extremely small proportion of the overall data (< 0.01%) and are not anticipated to significantly affect the results of this analysis.

However, addressing these missing values is essential for ensuring the consistency and reliability of our dataset for exploratory data analysis (EDA) and statistical evaluation. The following strategies were applied for handling the missing data:

1. ProfileName Missing Values:
    - The ProfileName column contains non-critical metadata that is not a direct factor in our helpfulness analysis. As such, missing values were replaced with empty string to maintain data completeness while avoiding unnecessary deletion of rows.
2. Summary Missing Values:
    - The Summary column provides a brief caption or overview of the review text. Missing entries in this column were filled with empty string to ensure uniformity.

By retaining these columns and addressing their missing values, the integrity of the overall dataset is preserved without introducing bias or skewness into the analysis. This approach ensures that all records remain available for subsequent data exploration and visualization.

After these modifications, an additional check for missing values was performed, confirming that the dataset is now complete and ready for statistical and exploratory analysis.



```
In [111]: # Fill missing values with empty string in 'ProfileName' and 'Summary' columns
          import pandas as pd

          df['ProfileName'] = df['ProfileName'].fillna('')
          df['Summary'] = df['Summary'].fillna('')
```

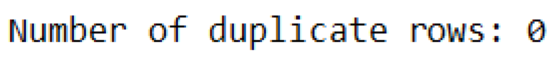
```
In [112]: #Checking null values after handeling 'ProfileName' and 'Summary' columns problem

          df.isnull().sum()

Out[112]: Id                        0
          ProductId                 0
          UserId                    0
          ProfileName               0
          HelpfulnessNumerator      0
          HelpfulnessDenominator    0
          Score                     0
          Time                      0
          Summary                   0
          Text                      0
          dtype: int64
```

Figure 3.1 Dataset columns after filling and checking

To ensure the integrity and reliability of the dataset, a check for duplicate rows was performed. Duplicate rows can introduce bias into the statistical analysis, distort data distributions, and inflate the significance of certain trends in the data. Therefore, identifying and removing duplicates is a critical preprocessing step

.

The analysis that is shown above revealed that the dataset does not contain any duplicate rows, ensuring that each review contributes unique information to the analysis.



```
Number of duplicate rows: 0
```

Figure 3.2 **:** Number of duplicate rows

The dataset consists of product reviews with a wide range of scores (Score) that typically range from 1 to 5 stars. To understand how these scores are distributed across the dataset, the Score column was visualized, and a breakdown of the number of reviews for each score was generated. This analysis provides a foundation for exploring the relationship between review scores and helpfulness metrics.

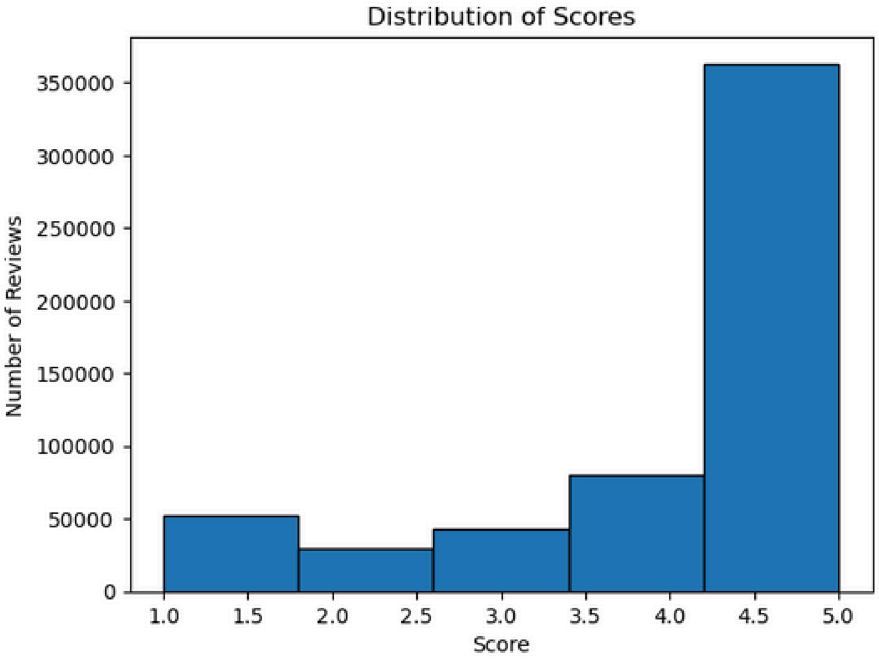


Figure 3.3 : Distribution of Score based on Number of Reviews

From the distribution above, it can be observed how user ratings are concentrated across different score groups.
This step is not only vital for understanding user sentiment patterns but also provides context for deeper analysis of how review scores influence helpfulness (helpfulness_ratio).

Reviews with higher scores may align with higher helpfulness_ratio due to generally positive feedback, while lower scores might show different trends.
This breakdown of Score distribution will be further investigated when testing relationships between numerical features (Score, word_count, review_length) and helpfulness_ratio.The selected features align with prior research on review helpfulness modeling (Mudambi & Schuff, 2010; Ghose & Ipeirotis, 2011).

3.2 Methodology

The methodology section outlines the systematic processes and techniques used to analyze and understand the helpfulness of product reviews from the Amazon Fine Food Reviews dataset. The approach is based on a series of structured steps designed to ensure high-quality data preparation, thorough exploratory data analysis (EDA), and interpretable statistical modeling. The overall goal is to extract valuable insights from customer feedback, focusing on factors that influence review helpfulness and identifying key trends.

The methodology follows a structured sequence:

1. Data Collection and Preprocessing: Ensures clean and complete data, with a focus on handling missing values, removing inconsistencies, and engineering features such as helpfulness_ratio, word_count, and review_length.
2. Exploratory Data Analysis (EDA): Provides insights into the dataset using descriptive statistics, visualizations, and distributions to identify patterns and relationships among features.
3. Correlation and Hypothesis Testing: Uses statistical techniques like Pearson correlation, T-tests, and ANOVA to investigate significant relationships and differences across groups of reviews.
4. Predictive Modeling with Linear Regression: Implements a simple and interpretable linear regression model to quantify how review features (e.g., length, score, and year) impact the helpfulness ratio.
5. Visualization of Results: Employs visual tools to represent distributions, trends, correlations, and modeling outcomes, ensuring all findings are interpretable and actionable.

Finally, all steps are performed with an emphasis on reproducibility, using standardized libraries and techniques to ensure consistency in results.

The flow of the methodology is illustrated in the diagram below, which presents the key steps in the research process.

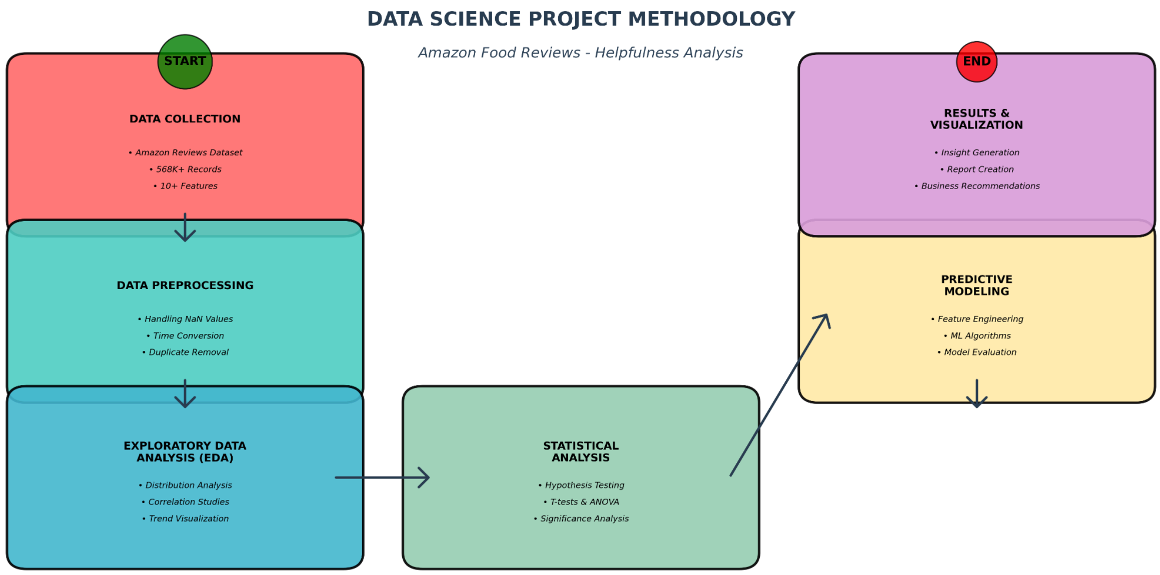


Figure 3.4 : Flow of steps used in the research process

3.3.Technologies and Tools

The implementation of this research was conducted using the Python programming language, leveraging its versatility and extensive ecosystem of libraries for data analysis, statistical modeling, and visualization. Python's simplicity and readability make it ideal for conducting reproducible research, enabling seamless exploration and manipulation of large datasets like the Amazon Fine Food Reviews dataset.

The analysis was performed in the Jupyter Notebook environment, an interactive computational platform that allows for the combination of code execution, data visualization, and narrative text in a single document.

3.3.1 Python Programming Language

Python is a high-level, dynamic programming language that emphasizes simplicity and readability, making it an ideal choice for conducting reproducible research in Data Science. Its flexibility supports multiple programming paradigms such as object-oriented, imperative, and functional programming, enabling efficient handling and analysis of large datasets. Python’s syntax is clear and concise, allowing developers to focus on solving problems rather than managing complex syntax. Despite being interpreted, Python’s extensive ecosystem of libraries offers robust functionality for data manipulation, statistical analysis, and visualization, which makes it widely popular in the realm of Data Science. Additionally, Python can integrate seamlessly with other tools and languages, such as databases like MySQL or high-performance algorithms in C++, further enhancing its applicability.

Throughout this research, Python was employed for:

- Data preprocessing (handling missing values, engineering features).
- Exploratory data analysis (EDA) and statistical testing (correlations, hypothesis testing).
- Creating graphical representations of insights using visualization libraries like Matplotlib.

3.3.2 Jupyter Notebook

Jupyter Notebook is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text (Dataquest Team, 2020). It is widely used in data science,machine learning , NLP and others for conducting interactive data analysis and creating reproducible research. Notebooks are composed of cells, which can either contain executable code or formatted text (using Markdown), making it easy to document and explain code. One of the powerful features of Jupyter is the ability to use "magics," which are special commands that extend its functionality, such as inline plotting with %matplotlib inline or measuring code performance with %%timeit. Additionally, Jupyter supports interactive widgets, enabling dynamic and interactive visualizations. Notebooks can be shared easily by saving them in a `.ipynb` file format or by using platforms like Binder or Google Colab (Dataquest Team, 2020).

3.3.3 Pandas

Pandas is an open-source Python library developed by Wes McKinney in 2008, enhancing Python's data analysis capabilities (Dataquest Team, 2020). It is designed for efficient handling of tabular, heterogeneous data and integrates well with libraries like NumPy and Matplotlib. Pandas provides tools for operations like reshaping data, handling time series, and performing statistical calculations such as mean and median. Its DataFrame object is central for creating structured datasets and exporting data in various formats, including CSV and Excel. Pandas is widely used across industries for tasks such as data cleaning, analysis, and modeling (Gupta & Bagchi, 2024). Throughout this research, the Pandas library was employed to efficiently handle data stored in CSV files, using DataFrames to organize and manipulate the data. It facilitated data cleaning by removing NaN values and unnecessary columns, as well as transforming the data into a usable format. These operations helped streamline the data preparation process, enabling more accurate and meaningful insights throughout the research.

3.3.4 **Matplotlib**

Matplotlib is a powerful Python library used for creating static, interactive, and animated visualizations (Saabith et al. 2021). It supports a wide range of plot types such as line charts, scatter plots, histograms, and bar charts. Matplotlib's flexibility allows it to integrate with various backends, providing diverse output formats suitable for different platforms. Its object-oriented API facilitates the embedding of plots into applications. Matplotlib is particularly valuable for data visualization tasks such as visualizing correlations, confidence intervals, and data distributions for deeper insights (Saabith et al. 2021).

In this project, Matplotlib is used to visually represent key insights from the Helpfulness Analysis & Prediction of product reviews. It effectively illustrates the percentage of missing data in features and the distribution of review scores, enabling a clearer understanding of data quality and trends.

3.3.5 Scikit-learn

Scikit-learn is a powerful Python library used for data science, providing simple and efficient tools for data analysis (Pedregosa et al. 2011). Scikit-learn includes utilities for model evaluation and selection, making it easy to tune and validate models. It is built on top of NumPy, SciPy, and matplotlib, ensuring high performance and integration with other scientific libraries. Widely used in both academic research and industry applications, Scikit-learn simplifies data science workflows significantly (Pedregosa et al. 2011).

Scikit-learn was used in this project for :

1. Regression modeling to analyze the quantitative impact of features like review length, score, and time on helpfulness.
2. Performance metrics (e.g., R^2 and mean squared error) to evaluate the predictive regression models.
3. Implementation of interpretable linear models that predict helpfulness_ratio.

3.3.6 NumPy

NumPy is a foundational Python library for numerical computing, offering efficient array operations and mathematical functions. In this project, NumPy was used for handling vectorized operations and performing matrix calculations essential for statistical and regression analysis.

3.3.7 Random

The Python random module provides tools for generating pseudo-random numbers, sampling data, and simulating probabilistic processes. It was used in this research for tasks such as random splitting of dataset into training and testing sets to ensure reproducibility of results.

3.3.8 SciPy

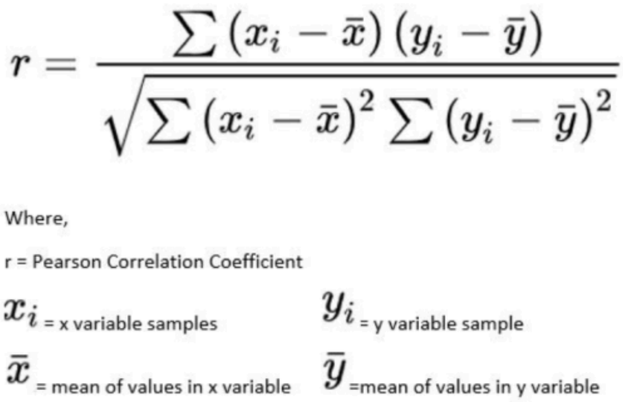
SciPy is a scientific computing library that extends NumPy's capabilities, providing functions for linear algebra, optimization, and integration. In this research, SciPy was used for advanced statistical analysis, such as computing correlations and performing hypothesis testing efficiently.

3.4 Theoretical Foundations of the Methodology

This section outlines the theoretical foundations that guide the methodology used in this project. By exploring key concepts in statistical analysis, linear modeling, and data exploration, this section provides a solid framework for understanding the approach taken to analyze and predict helpfulness metrics in product reviews.

3.4.1 Pearson Correlation Coefficient

The Pearson correlation coefficient measures the strength and direction of the linear relationship between two numerical variables. For this project, it helps identify how numerical attributes (e.g., score, word_count, and review_length) impact helpfulness_ratio. This metric is ideal due to its simplicity for capturing relationships.



$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Where,

r = Pearson Correlation Coefficient

x_i = x variable samples y_i = y variable sample

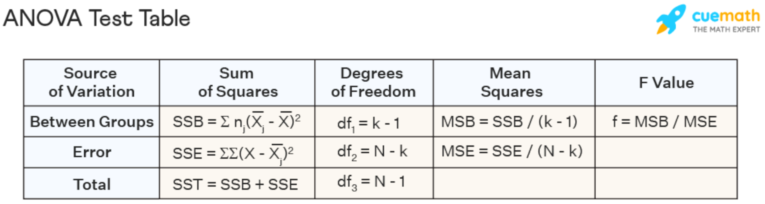
$\bar{x}$ = mean of values in x variable $\bar{y}$ =mean of values in y variable

Figure 3.5 Pearson correlation coefficient formula

3.4.2 T-tests and ANOVA

○ T-tests are used to determine significant mean differences in helpfulness_ratio between binary groups (e.g., positive vs negative reviews based on score thresholds).

○ ANOVA (Analysis of Variance) is applied to analyze variance across multiple groups, such as examining how different Score categories affect helpfulness_ratio.

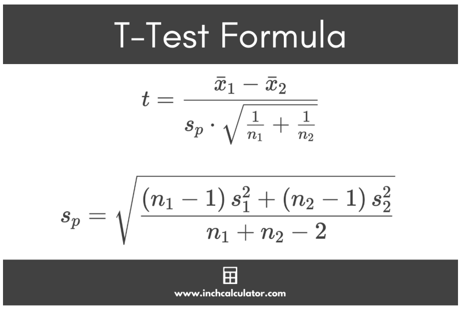


ANOVA Test Table

cuemath THE MATH EXPERT

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Squares	F Value
Between Groups	SSB = $\Sigma n_j(\bar{X}_j - \bar{X})^2$	$df_1 = k - 1$	MSB = SSB / (k - 1)	f = MSB / MSE
Error	SSE = $\Sigma\Sigma(X - \bar{X}_j)^2$	$df_2 = N - k$	MSE = SSE / (N - k)	
Total	SST = SSB + SSE	$df_3 = N - 1$		

Figure 3.4 ANOVA Test table formulas



T-Test Formula

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s_p \cdot \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

$$s_p = \sqrt{\frac{(n_1 - 1)\, s_1^2 + (n_2 - 1)\, s_2^2}{n_1 + n_2 - 2}}$$

www.inchcalculator.com

Figure 3.6 T-Test formula

3.4.3 Linear Regression for Predictive Analysis

Linear regression provides a straightforward method for predicting the helpfulness ratio (helpfulness_ratio) based on explanatory features such as review_length, score, and year. The regression model operates under the following formula:

$$Y=\beta0+\beta1X1+\beta2X2+\cdots+\beta nXn+\varepsilon$$

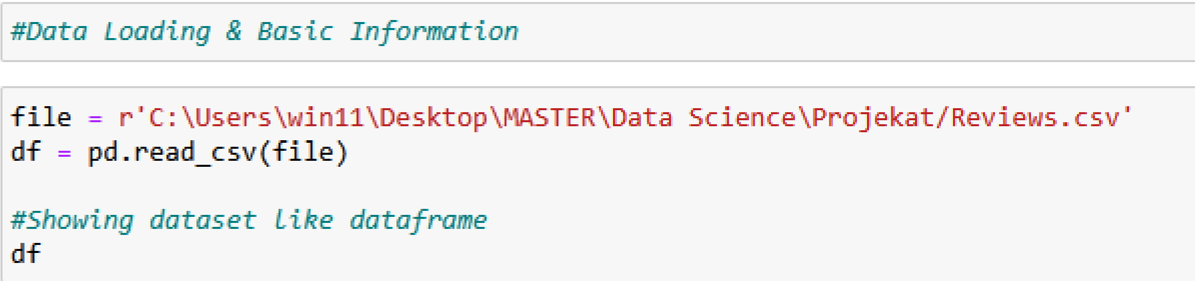
Where Y is the target variable, X1,X2,…,Xn represent feature variables, β0 is the intercept, β1,…,βn are coefficients, and ε is the error term.Linear regression is chosen in this research for its interpretability, making it ideal for analyzing the quantitative influence of multiple review features.

3.5 Implementation

This section outlines the step-by-step approach used to analyze helpfulness metrics and explore patterns in product reviews, emphasizing the importance of each phase in the overall Data Science workflow. The process begins with data collection and loading the dataset into the environment for analysis. This is followed by data preprocessing to clean, normalize, and prepare the data, ensuring readiness for further exploration.

3.5.1 Data Loading

To begin the data analysis process, the dataset is loaded using Pandas. The dataset Reviews.csv contains a collection of reviews with multiple attributes such as ProductId, UserId, Score, Text

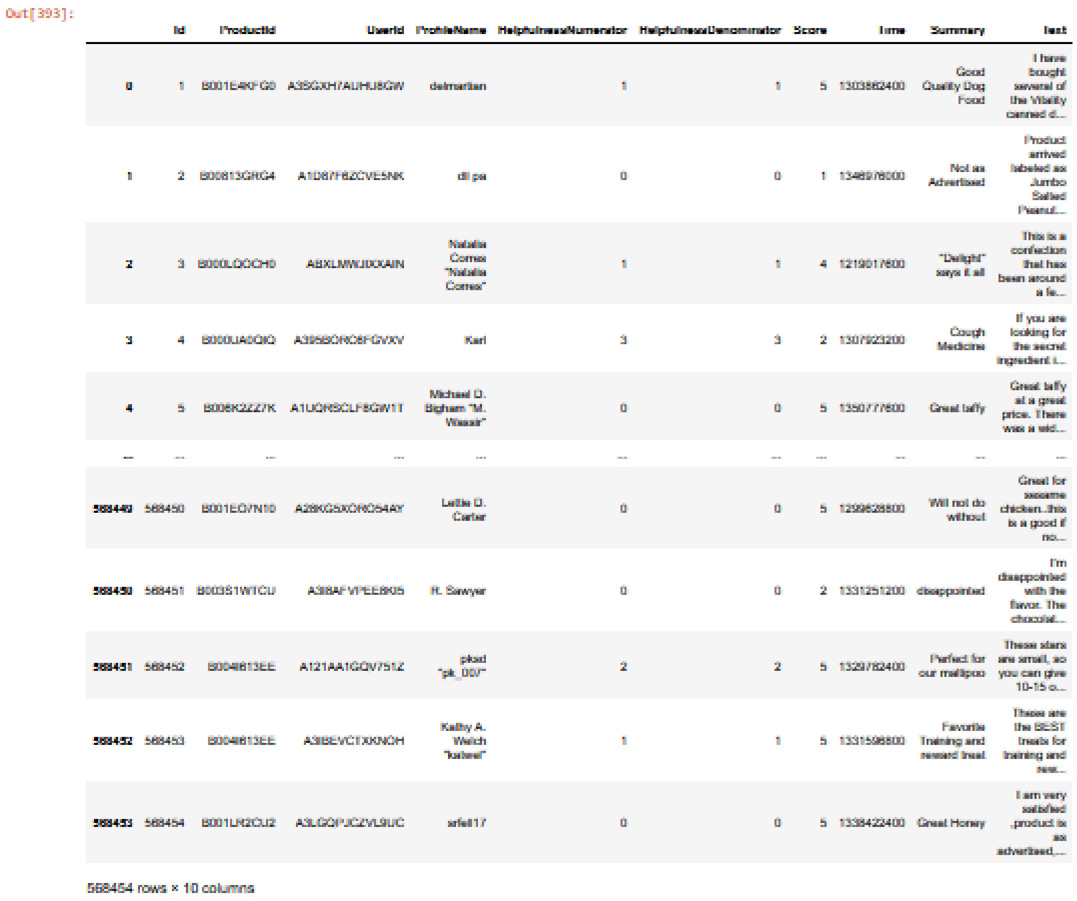


```
#Data Loading & Basic Information
```

```
file = r'C:\Users\win11\Desktop\MASTER\Data Science\Projekat/Reviews.csv'
df = pd.read_csv(file)

#Showing dataset like dataframe
df
```

Figure 3.7 Dataset Loading

After loading the dataset, it is useful to inspect the dataset dataframe to understand its structure:

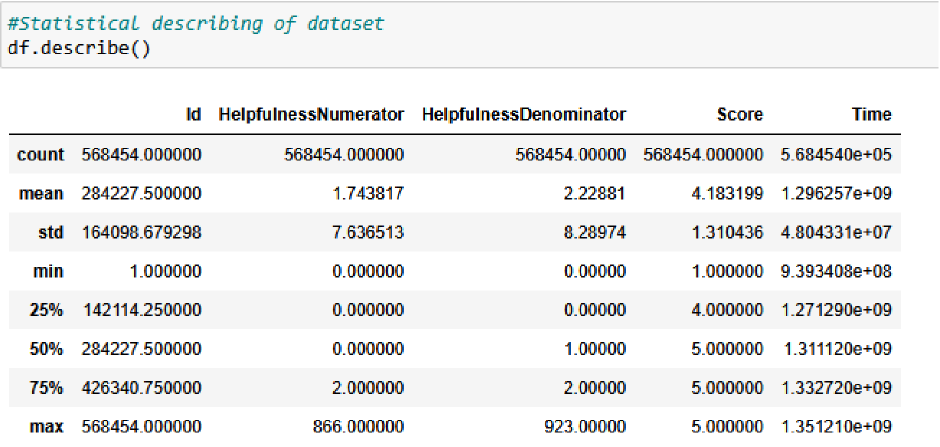


Out[393]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...
...	...	...	...	...	...	...	...	...	...	...
568449	568450	B001EO7N10	A28KG5XORO54AY	Lettie D. Carter	0	0	5	1299628800	Will not do without	Great for sesame chicken..this is a good if no...
568450	568451	B003S1WTCU	A3I8AFVPEE8KI5	R. Sawyer	0	0	2	1331251200	disappointed	I'm disappointed with the flavor. The chocolat...
568451	568452	B004I613EE	A121AA1GQV751Z	pksd "pk_007"	2	2	5	1329782400	Perfect for our maltipoo	These stars are small, so you can give 10-15 o...
568452	568453	B004I613EE	A3IBEVCTXKNOH	Kathy A. Welch "katwel"	1	1	5	1331596800	Favorite Training and reward treat	These are the BEST treats for training and rew...
568453	568454	B001LR2CU2	A3LGQPJCZVL9UC	srfell17	0	0	5	1338422400	Great Honey	I am very satisfied ,product is as advertised,...

568454 rows × 10 columns

Figure 3.8 Dataset DataFrame

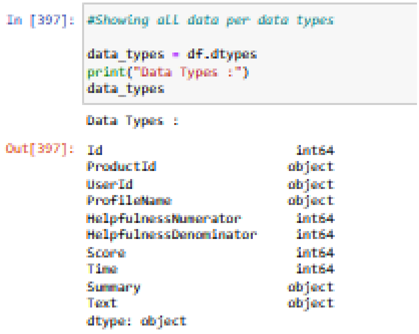


```
#Statistical describing of dataset
df.describe()
```

	Id	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time
count	568454.000000	568454.000000	568454.00000	568454.000000	5.684540e+05
mean	284227.500000	1.743817	2.22881	4.183199	1.296257e+09
std	164098.679298	7.636513	8.28974	1.310436	4.804331e+07
min	1.000000	0.000000	0.00000	1.000000	9.393408e+08
25%	142114.250000	0.000000	0.00000	4.000000	1.271290e+09
50%	284227.500000	0.000000	1.00000	5.000000	1.311120e+09
75%	426340.750000	2.000000	2.00000	5.000000	1.332720e+09
max	568454.000000	866.000000	923.00000	5.000000	1.351210e+09

Figure 3.9 Dataset Statistic

This allows for a quick overview of the data, including column names, sample data and overall statistics of dataset. The next thing is overview of dataset to see columns number and it's type :



```
In [397]: #Showing all data per data types

          data_types = df.dtypes
          print("Data Types :")
          data_types

          Data Types :

Out[397]: Id                           int64
          ProductId                   object
          UserId                      object
          ProfileName                 object
          HelpfulnessNumerator         int64
          HelpfulnessDenominator       int64
          Score                        int64
          Time                         int64
          Summary                     object
          Text                        object
          dtype: object
```

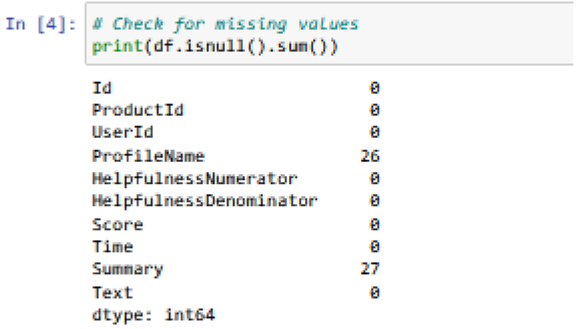
Figure 3.10 Dataset data types

This initial inspection ensures that the data is properly loaded and provides an overview of its completeness and structure, helping to identify any potential data issues before proceeding with further analysis or preprocessing steps.

3.5.2 Data Preprocessing

Data preprocessing is a crucial step in the data analysis pipeline, as it ensures that the dataset is clean, consistent, and ready for analysis or modeling. This phase typically involves tasks such as handling missing values, encoding categorical variables, normalizing numerical data, and removing unnecessary or irrelevant features. The following code outlines the preprocessing steps applied to the dataset to prepare it for further analysis.

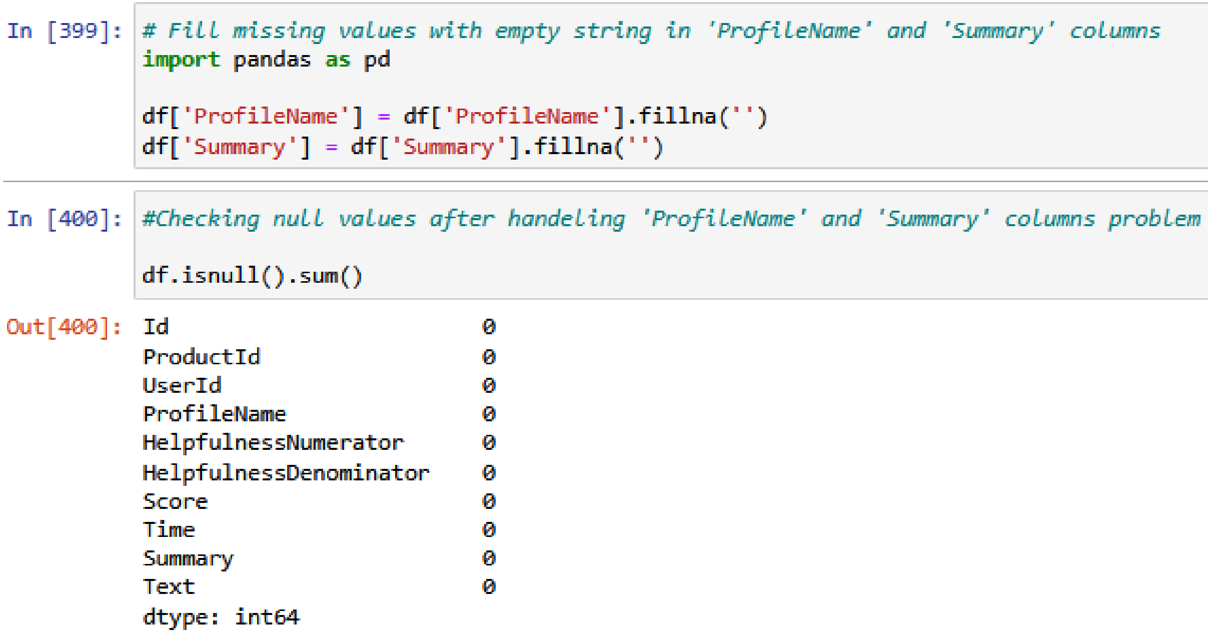
The code checks for missing values and visualizes the percentage of missing data for each feature. This helps to assess the quality of the dataset and decide whether to drop or impute missing values.



```
In [4]: # Check for missing values
        print(df.isnull().sum())

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

Figure 3.11 Checking for missing values

The ‘ProfileName’and ‘Summary' columns are filled with empty string for easier handling during working with those columns in the future phases.

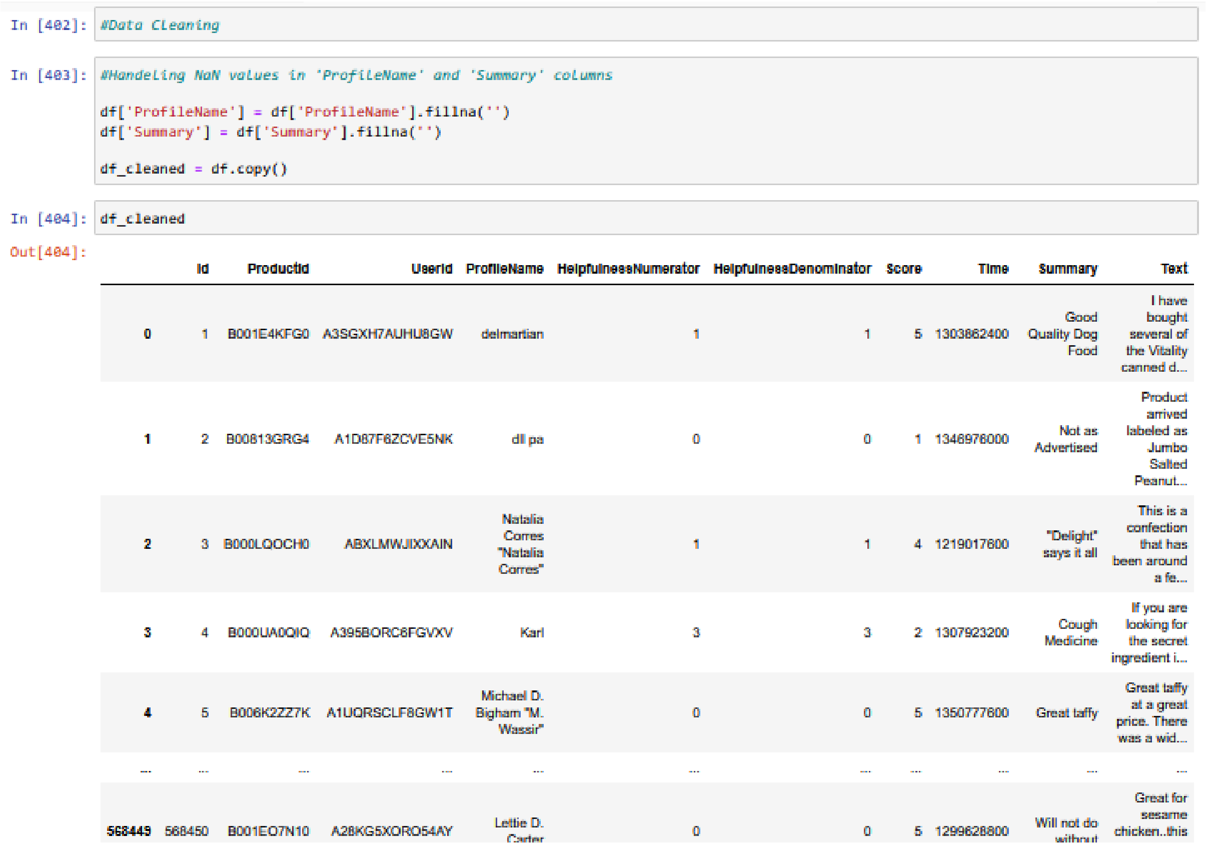


```
In [399]: # Fill missing values with empty string in 'ProfileName' and 'Summary' columns
          import pandas as pd

          df['ProfileName'] = df['ProfileName'].fillna('')
          df['Summary'] = df['Summary'].fillna('')

In [400]: #Checking null values after handeling 'ProfileName' and 'Summary' columns problem

          df.isnull().sum()

Out[400]: Id                        0
          ProductId                 0
          UserId                    0
          ProfileName               0
          HelpfulnessNumerator      0
          HelpfulnessDenominator    0
          Score                     0
          Time                      0
          Summary                   0
          Text                      0
          dtype: int64
```

Figure 3.12 ‘ProfileName’ and ‘Summary’ columns modification

3.5.3 Data Cleaning

After performing the data preprocessing process, the next phase is to deal with data cleaning. Initially , datasets always come in the form of sql,csv and other unstructured data formats. Data cleaning in this research focuses on ensuring numerical consistency, handling missing values, removing invalid helpfulness entries, and eliminating duplicate records. Since the textual content was not subjected to natural language processing, no HTML tag removal or text normalization was applied.The picture above shows the process of data cleaning and the dataset dataframe after cleaning.

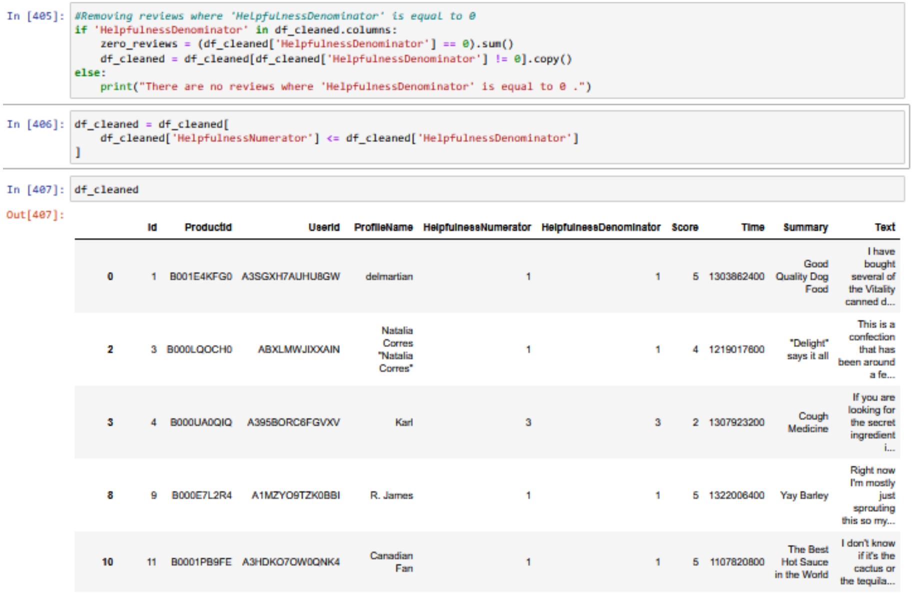


```
In [402]: #Data Cleaning

In [403]: #Handeling NaN values in 'ProfileName' and 'Summary' columns

          df['ProfileName'] = df['ProfileName'].fillna('')
          df['Summary'] = df['Summary'].fillna('')

          df_cleaned = df.copy()

In [404]: df_cleaned
Out[404]:
```

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...
...	...	...	...	...	...	...	...	...	...	...
568449	568450	B001EO7N10	A28KG5XORO54AY	Lettie D. Carter	0	0	5	1299628800	Will not do without	Great for sesame chicken..this

Figure 3.13 Data cleaning - before and after

Since the entire topic of this research is based on Helpfulness Analysis & Prediction, when cleaning data, it is also necessary to pay attention to the 'Helpfulness Denominator' columns because they represent the starting columns for further work on this research.

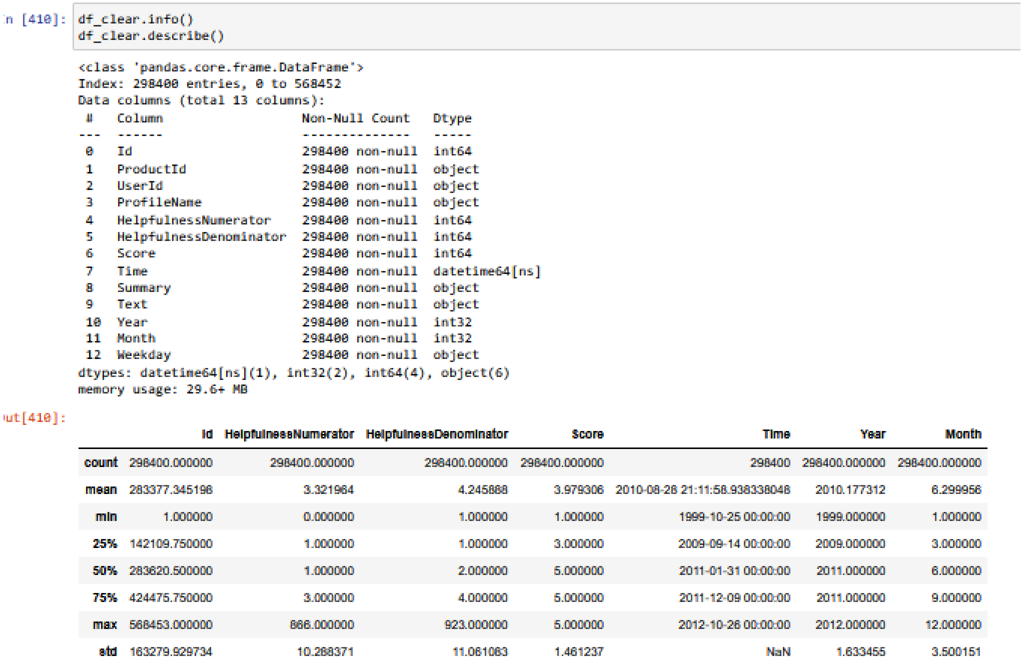


```
In [405]: #Removing reviews where 'HelpfulnessDenominator' is equal to 0
          if 'HelpfulnessDenominator' in df_cleaned.columns:
              zero_reviews = (df_cleaned['HelpfulnessDenominator'] == 0).sum()
              df_cleaned = df_cleaned[df_cleaned['HelpfulnessDenominator'] != 0].copy()
          else:
              print("There are no reviews where 'HelpfulnessDenominator' is equal to 0 .")

In [406]: df_cleaned = df_cleaned[
              df_cleaned['HelpfulnessNumerator'] <= df_cleaned['HelpfulnessDenominator']
          ]

In [407]: df_cleaned
Out[407]:
```

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
8	9	B000E7L2R4	A1MZYO9TZK0BBI	R. James	1	1	5	1322006400	Yay Barley	Right now I'm mostly just sprouting this so my...
10	11	B0001PB9FE	A3HDKO7OW0QNK4	Canadian Fan	1	1	5	1107820800	The Best Hot Sauce in the World	I don't know if it's the cactus or the tequila...

Figure 3.14 Removing reviews where 'HelpfulnessDenominator' is equal to 0

After total data cleaning, as well as transformation of the dataset during cleaning ('Text' column in datatime, removal of duplicates after cleaning), significant changes are observed both in the dataset itself and in its statistics.



```
n [410]: df_clear.info()
         df_clear.describe()

<class 'pandas.core.frame.DataFrame'>
Index: 298400 entries, 0 to 568452
Data columns (total 13 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   Id                      298400 non-null  int64
 1   ProductId               298400 non-null  object
 2   UserId                  298400 non-null  object
 3   ProfileName             298400 non-null  object
 4   HelpfulnessNumerator    298400 non-null  int64
 5   HelpfulnessDenominator  298400 non-null  int64
 6   Score                   298400 non-null  int64
 7   Time                    298400 non-null  datetime64[ns]
 8   Summary                 298400 non-null  object
 9   Text                    298400 non-null  object
 10  Year                    298400 non-null  int32
 11  Month                   298400 non-null  int32
 12  Weekday                 298400 non-null  object
dtypes: datetime64[ns](1), int32(2), int64(4), object(6)
memory usage: 29.6+ MB

ut[410]:
```

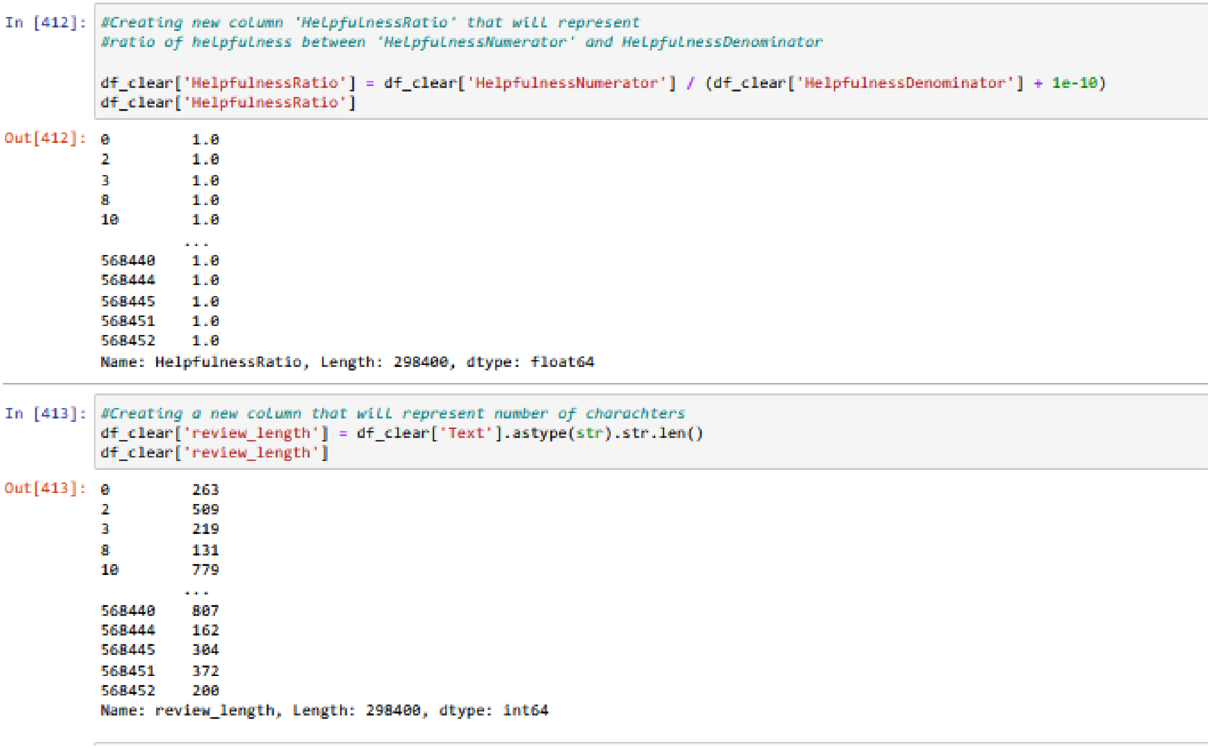
	Id	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Year	Month
count	298400.000000	298400.000000	298400.000000	298400.000000	298400	298400.000000	298400.000000
mean	283377.345196	3.321964	4.245888	3.979306	2010-08-28 21:11:58.938338048	2010.177312	6.299956
min	1.000000	0.000000	1.000000	1.000000	1999-10-25 00:00:00	1999.000000	1.000000
25%	142109.750000	1.000000	1.000000	3.000000	2009-09-14 00:00:00	2009.000000	3.000000
50%	283620.500000	1.000000	2.000000	5.000000	2011-01-31 00:00:00	2011.000000	6.000000
75%	424475.750000	3.000000	4.000000	5.000000	2011-12-09 00:00:00	2011.000000	9.000000
max	568453.000000	866.000000	923.000000	5.000000	2012-10-26 00:00:00	2012.000000	12.000000
std	163279.929734	10.288371	11.061083	1.461237	NaN	1.633455	3.500151

Figure 3.15 Dataset and its statistics after cleaning

3.5.4 Feature engineering

In order to even start working with concrete analyses and statistical calculations, the first thing is to determine feature engineering and later on define the research question.

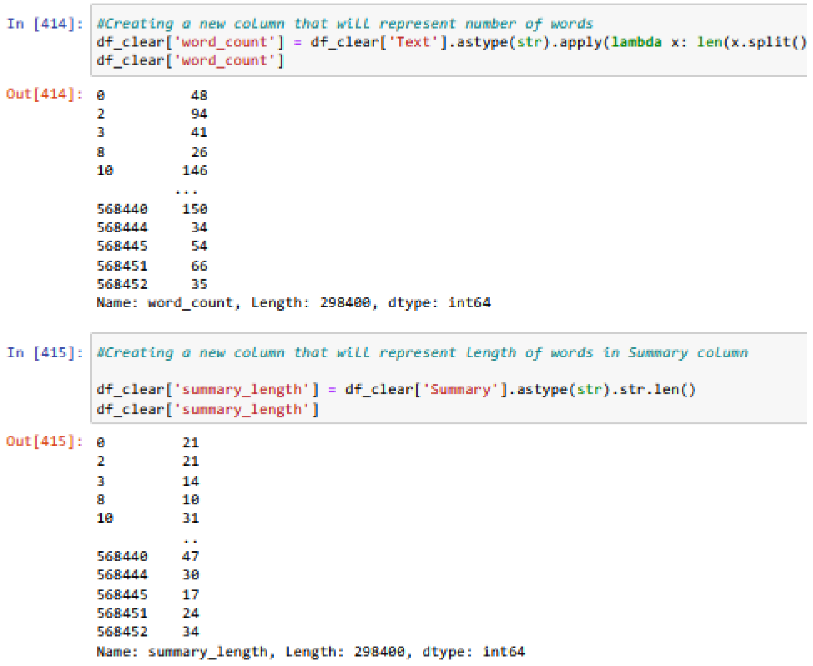
Due to statistical calculations, it is necessary to create new columns within the already existing dataset. Creating a new column 'HelpfulnessRatio' that will represent the ratio of helpfulness between 'HelpfulnessNumerator' and 'HelpfulnessDenominator'. This column is important because the goal of this research is the Helpfulness analysis of reviews.



```
In [412]: #Creating new column 'HelpfulnessRatio' that will represent
          #ratio of helpfulness between 'HelpfulnessNumerator' and HelpfulnessDenominator

          df_clear['HelpfulnessRatio'] = df_clear['HelpfulnessNumerator'] / (df_clear['HelpfulnessDenominator'] + 1e-10)
          df_clear['HelpfulnessRatio']

Out[412]: 0         1.0
          2         1.0
          3         1.0
          8         1.0
          10        1.0
                   ...
          568440    1.0
          568444    1.0
          568445    1.0
          568451    1.0
          568452    1.0
          Name: HelpfulnessRatio, Length: 298400, dtype: float64

In [413]: #Creating a new column that will represent number of charachters
          df_clear['review_length'] = df_clear['Text'].astype(str).str.len()
          df_clear['review_length']

Out[413]: 0         263
          2         509
          3         219
          8         131
          10        779
                   ...
          568440    807
          568444    162
          568445    304
          568451    372
          568452    200
          Name: review_length, Length: 298400, dtype: int64
```

Figure 3.16 Creating new columns 'HelpfulnessRatio' and ‘review_length’

Next, the 'review_length' and 'word_count' columns are created, which represent the length of the review and the number of words in it.



```
In [414]: #Creating a new column that will represent number of words
          df_clear['word_count'] = df_clear['Text'].astype(str).apply(lambda x: len(x.split()
          df_clear['word_count']

Out[414]: 0          48
          2          94
          3          41
          8          26
          10        146
                   ...
          568440    150
          568444     34
          568445     54
          568451     66
          568452     35
          Name: word_count, Length: 298400, dtype: int64

In [415]: #Creating a new column that will represent length of words in Summary column

          df_clear['summary_length'] = df_clear['Summary'].astype(str).str.len()
          df_clear['summary_length']

Out[415]: 0          21
          2          21
          3          14
          8          10
          10         31
                     ..
          568440     47
          568444     30
          568445     17
          568451     24
          568452     34
          Name: summary_length, Length: 298400, dtype: int64
```

Figure 3.17 Creating new columns ‘word_count’ and ‘summary_length’

4. Results and Discussion

4.1 Introduction to the three main research objectives

This research is structured around three primary analytical objectives:

1. Analysis of Rating Distributions and Helpfulness Patterns

This objective examines how product ratings (Score) are distributed over time and how they relate to perceived review helpfulness (HelpfulnessRatio).

2. Statistical Evaluation of Factors Influencing Review Helpfulness

This objective investigates the relationships between review helpfulness and selected numerical features, including word_count, review_length, score, and temporal attributes, using correlation analysis, hypothesis testing, and ANOVA.

3. Predictive Analysis of Review Helpfulness Using Linear Regression

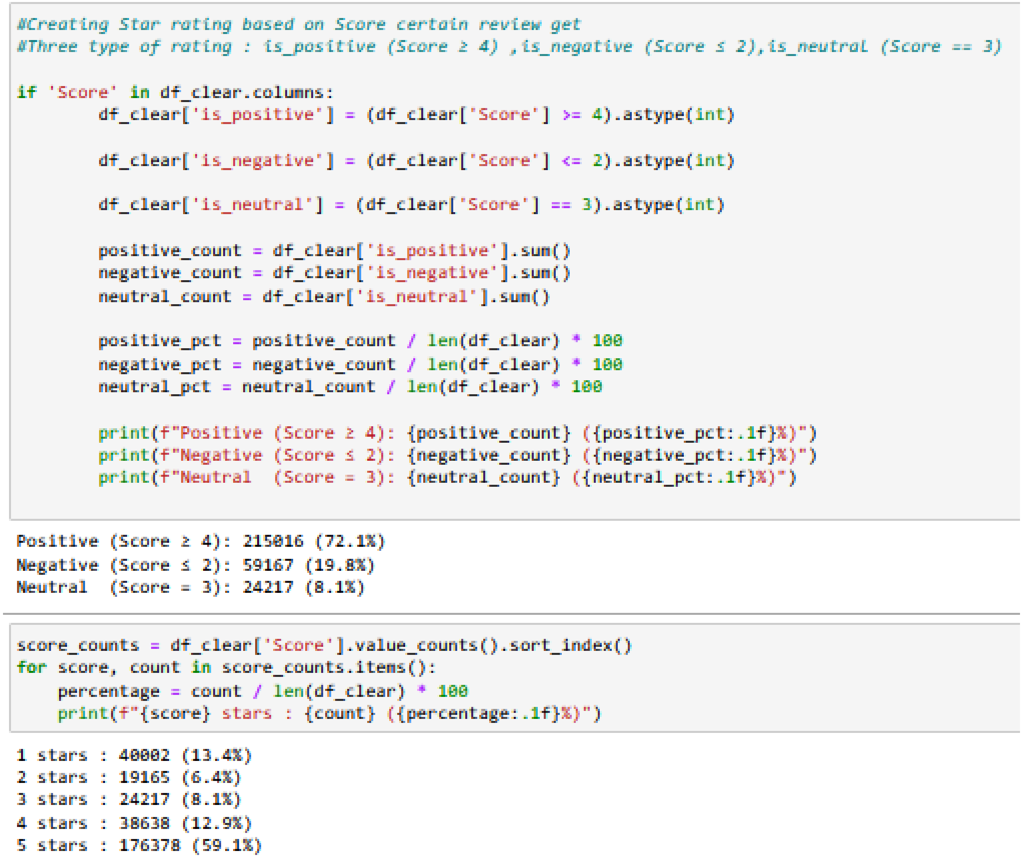
This objective evaluates the extent to which selected features can explain variations in HelpfulnessRatio using an interpretable linear regression model.

4.2 Exploratory Data Analysis (EDA)

4.2.1 Distribution of Review Scores

In assessing whether a given review is helpful or not, the most relied upon data is the ratings and scores of certain products. This dataset contains this information in the form of the 'Score' column.

Reviews (as well as the products they are written for) are classified according to their ratings: Positive (Score ≥ 4) Neutral (Score = 3) Negative (Score <=2).After grouping, we can also see the percentage share of these groups within the dataset itself.



```
#Creating Star rating based on Score certain review get
#Three type of rating : is_positive (Score ≥ 4) ,is_negative (Score ≤ 2),is_neutral (Score == 3)

if 'Score' in df_clear.columns:
        df_clear['is_positive'] = (df_clear['Score'] >= 4).astype(int)

        df_clear['is_negative'] = (df_clear['Score'] <= 2).astype(int)

        df_clear['is_neutral'] = (df_clear['Score'] == 3).astype(int)

        positive_count = df_clear['is_positive'].sum()
        negative_count = df_clear['is_negative'].sum()
        neutral_count = df_clear['is_neutral'].sum()

        positive_pct = positive_count / len(df_clear) * 100
        negative_pct = negative_count / len(df_clear) * 100
        neutral_pct = neutral_count / len(df_clear) * 100

        print(f"Positive (Score ≥ 4): {positive_count} ({positive_pct:.1f}%)")
        print(f"Negative (Score ≤ 2): {negative_count} ({negative_pct:.1f}%)")
        print(f"Neutral  (Score = 3): {neutral_count} ({neutral_pct:.1f}%)")
```

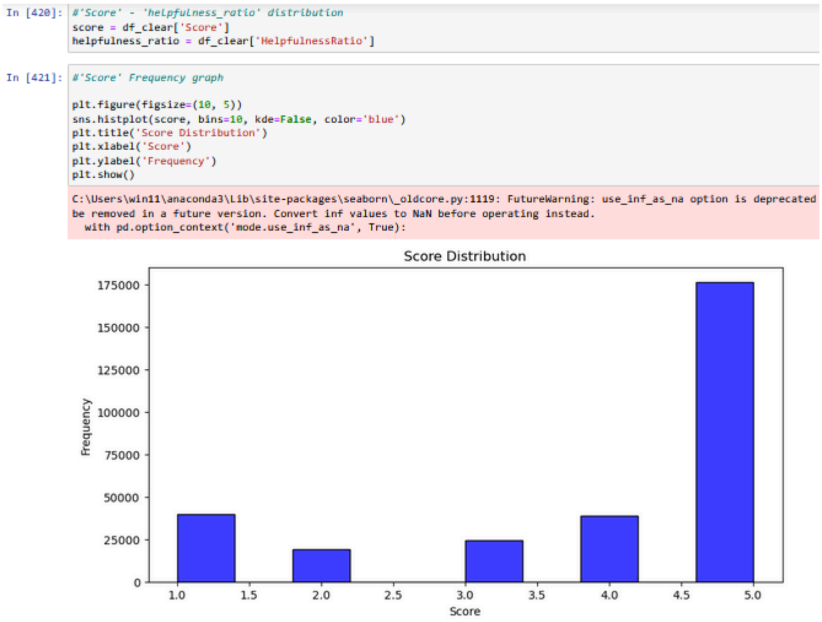
```
Positive (Score ≥ 4): 215016 (72.1%)
Negative (Score ≤ 2): 59167 (19.8%)
Neutral  (Score = 3): 24217 (8.1%)
```

```
score_counts = df_clear['Score'].value_counts().sort_index()
for score, count in score_counts.items():
    percentage = count / len(df_clear) * 100
    print(f"{score} stars : {count} ({percentage:.1f}%)")
```

```
1 stars : 40002 (13.4%)
2 stars : 19165 (6.4%)
3 stars : 24217 (8.1%)
4 stars : 38638 (12.9%)
5 stars : 176378 (59.1%)
```

Figure 4.1 Star rating and its percentages

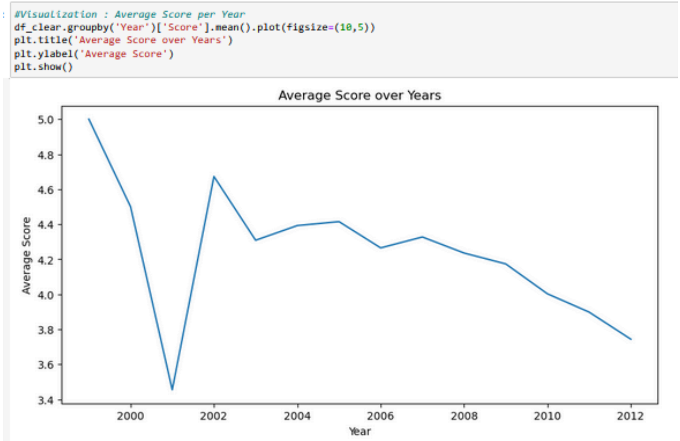
The picture below shows a graphic representation of the ratings and their frequencies within the dataset.



```
In [420]: #'Score' - 'helpfulness_ratio' distribution
          score = df_clear['Score']
          helpfulness_ratio = df_clear['HelpfulnessRatio']

In [421]: #'Score' Frequency graph

          plt.figure(figsize=(10, 5))
          sns.histplot(score, bins=10, kde=False, color='blue')
          plt.title('Score Distribution')
          plt.xlabel('Score')
          plt.ylabel('Frequency')
          plt.show()

          C:\Users\win11\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated
          be removed in a future version. Convert inf values to NaN before operating instead.
            with pd.option_context('mode.use_inf_as_na', True):
```

Figure 4.2 Star rating graphical representation

The scores of the reviews themselves can be monitored in different time intervals. Ratings can rise and fall, and the graph below represents the movement of the average rating over time.



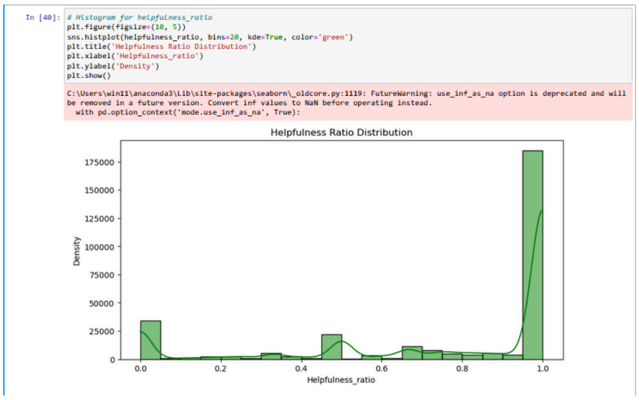
```
#Visualization : Average Score per Year
df_clear.groupby('Year')['Score'].mean().plot(figsize=(10,5))
plt.title('Average Score over Years')
plt.ylabel('Average Score')
plt.show()
```

Figure 4.3 Average Score per Year

We can observe that the average Score has been changed through time, having more or less fluctuations.

4.2.2 Helpfulness Ratio Distribution

The Helpfulness Ratio Distribution shows how users perceive the usefulness of reviews. It tells us what portion of reviews are considered helpful by other users and. Analyzing this distribution helps us understand the quality of reviews and user expectations. If most reviews have a high helpfulness ratio, it indicates that users find them valuable. This insight is essential for improving review systems and predicting helpfulness in the future. The above graph represents helpfulness ratio distribution and its density.

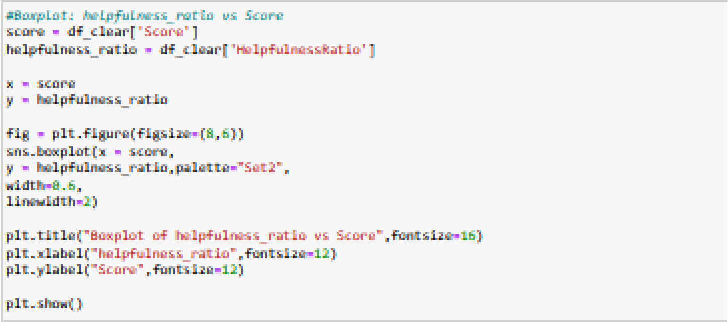


```
In [40]: # Histogram for helpfulness_ratio
         plt.figure(figsize=(10, 5))
         sns.histplot(helpfulness_ratio, bins=20, kde=True, color='green')
         plt.title('Helpfulness Ratio Distribution')
         plt.xlabel('Helpfulness_ratio')
         plt.ylabel('Density')
         plt.show()
```

```
C:\Users\win11\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will
be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Figure 4.4 Helpfulness Ratio Distribution

An important factor that affects the helpfulness ratio is the score. Since the helpfulness ratio represents the ratio between the helpfulness numerator and the helpfulness denominator, it is taken as a certain "measure" for understanding review dynamics and user decision-making processes.. It should be noted that both reviews (as well as their comments) and therefore most of the decisions made by future buyers of a product are based on the rating of the given product itself.



```
#Boxplot: helpfulness_ratio vs Score
score = df_clear['Score']
helpfulness_ratio = df_clear['HelpfulnessRatio']

x = score
y = helpfulness_ratio

fig = plt.figure(figsize=(8,6))
sns.boxplot(x = score,
y = helpfulness_ratio,palette="Set2",
width=0.6,
linewidth=2)

plt.title("Boxplot of helpfulness_ratio vs Score",fontsize=16)
plt.xlabel("helpfulness_ratio",fontsize=12)
plt.ylabel("Score",fontsize=12)

plt.show()
```

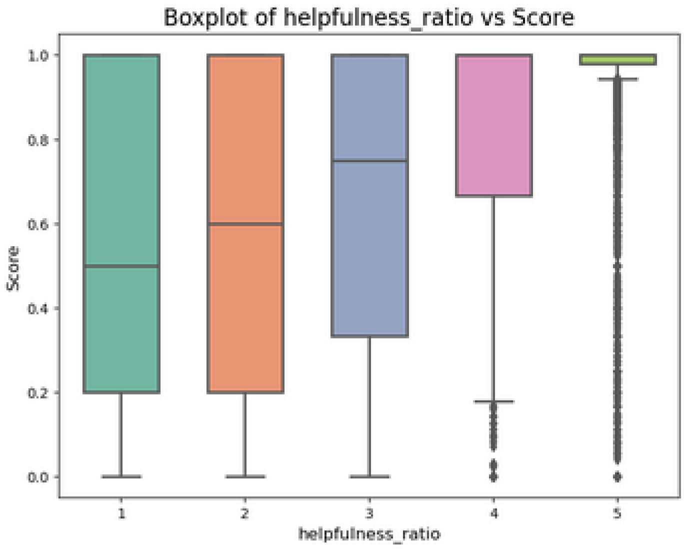
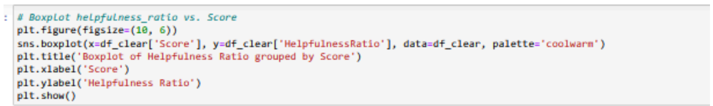


Figure 4.5 Boxplot : Helpfulness Ratio vs Score

Based on sentiment, we previously grouped reviews according to their scores. The groups range from 1 to 5 (the minimum rating a user can leave on a review is 1, and the maximum is 5). The helpfulness ratio was also established according to these groupings.



```
# Boxplot helpfulness_ratio vs. Score
plt.figure(figsize=(10, 6))
sns.boxplot(x=df_clear['Score'], y=df_clear['HelpfulnessRatio'], data=df_clear, palette='coolwarm')
plt.title('Boxplot of Helpfulness Ratio grouped by Score')
plt.xlabel('Score')
plt.ylabel('Helpfulness Ratio')
plt.show()
```

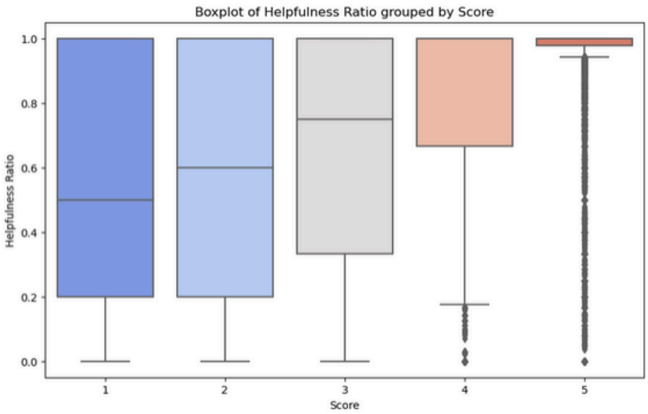


Figure 4.6 Boxplot : Helpfulness Ratio groped by Score

Based on all the graphs, groupings, plots, as well as the very basic use of the helpfulness ratio to prove the impact of reviews on future users, the analysis indicates a pronounced concentration of helpfulness ratios in proximity to 1.0 because this concentration is partly influenced by the dataset structure, as reviews with low helpfulness denominators were excluded during preprocessing. These results are consistent with earlier findings showing a strong association between rating extremity and review helpfulness (Mudambi & Schuff, 2010).

4.2.3 Correlation Analysis

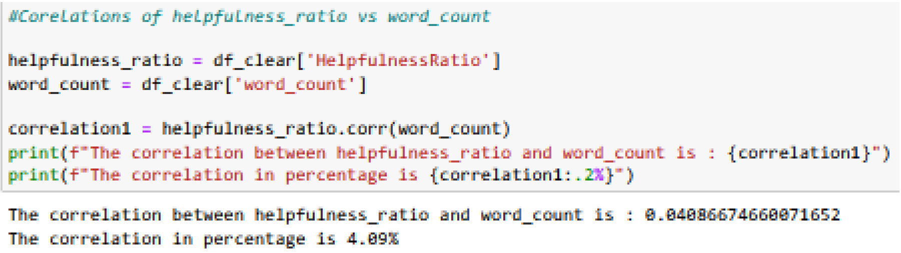
Correlation analysis is a statistical technique used to measure and analyze the strength and direction of a relationship between two or more variables. It provides insights into whether and how variables are related without establishing causation.Studies such as Forman et al. (2008) and Liu et al. (2007) have demonstrated that the identity of reviewers and review features contribute significantly to perceived helpfulness. This suggests that while numerical correlations provide valuable insights, other behavioral factors also play a critical role in shaping user perceptions of review usefulness.

Correlation is important for assessing the relationship between two groups. To analyze reviews and assess whether they are helpful or not, it is necessary to calculate the correlation between 3 already existing (created) columns: helpfulness_ratio , word_count , review_length .

4.2.4 Correlations of helpfulness_ratio vs word_count

Correlation between helpfulness_ratio and word_count represent one of earliest hypotheses in this research. Different factors can influence whether a review is helpful or not. One of them is the number of words in the review itself.

1. Hypothesis 1: The number of words in the review affects whether a review is helpful or not.
2. Hypothesis 2: The number of words in the review has no (or almost no) effect on whether a review is helpful or not.



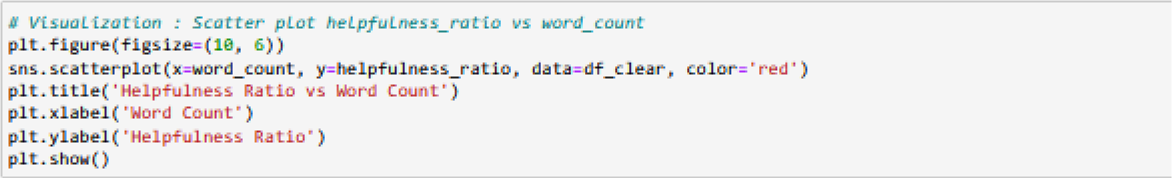
```
#Corelations of helpfulness_ratio vs word_count

helpfulness_ratio = df_clear['HelpfulnessRatio']
word_count = df_clear['word_count']

correlation1 = helpfulness_ratio.corr(word_count)
print(f"The correlation between helpfulness_ratio and word_count is : {correlation1}")
print(f"The correlation in percentage is {correlation1:.2%}")
```

```
The correlation between helpfulness_ratio and word_count is : 0.04086674660071652
The correlation in percentage is 4.09%
```

Figure 4.7 Correlation between helpfulness_ratio and word_count

A correlation result of 0.041 (or 4.09%) tells us that there is a very weak relationship between the number of words in review and its usefulness. This result indicates that:

1. The number of words in review does not significantly affect its usefulness
2. People do not value reviews by its length
3. There are other factors that affect the usefulness of reviews



```
# Visualization : Scatter plot helpfulness_ratio vs word_count
plt.figure(figsize=(10, 6))
sns.scatterplot(x=word_count, y=helpfulness_ratio, data=df_clear, color='red')
plt.title('Helpfulness Ratio vs Word Count')
plt.xlabel('Word Count')
plt.ylabel('Helpfulness Ratio')
plt.show()
```

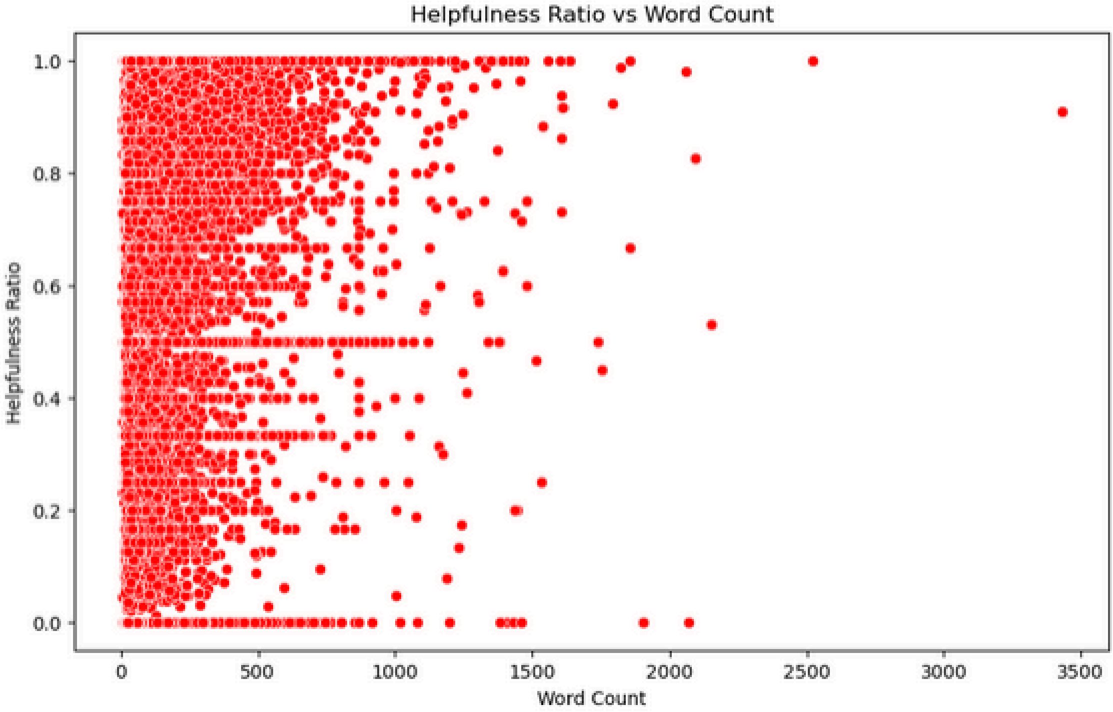


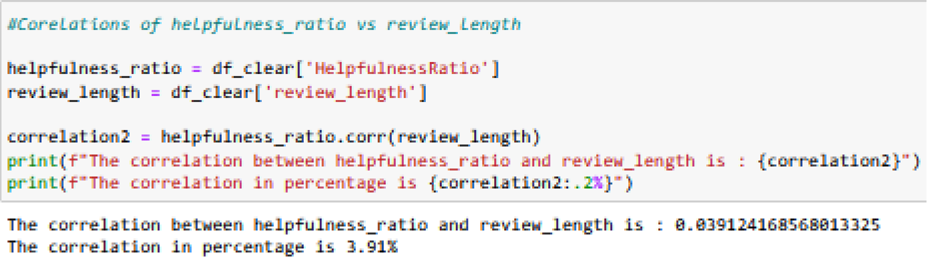
Figure 4.8 Scatter plot helpfulness_ratio vs word_count

Given the correlation coefficient (0.041 or 4.09%), there is insufficient evidence to reject the null hypothesis (H_0). The relationship between word_count and helpfulness_ratio is statistically negligible, suggesting that the length of a review in terms of word count has little to no impact on its perceived helpfulness. Other unexplored factors likely contribute to user perceptions of helpfulness.

4.2.5 Correlations of helpfulness_ratio vs review_length

Correlation between helpfulness_ratio and review_length represent the second of the earliest hypotheses in this research. Different factors can influence whether a review is helpful or not. One of them is the length of the review itself.

3. Hypothesis 1: The length of the review itself affects whether a review is helpful or not.
4. Hypothesis 2: The length of the review itself has no (or almost no) effect on whether a review is helpful or not.



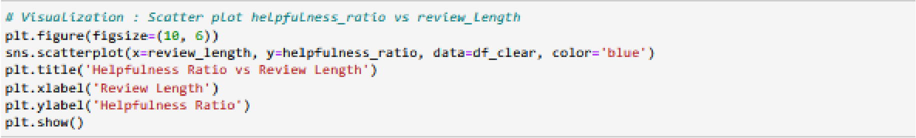
```
#Corelations of helpfulness_ratio vs review_length

helpfulness_ratio = df_clear['HelpfulnessRatio']
review_length = df_clear['review_length']

correlation2 = helpfulness_ratio.corr(review_length)
print(f"The correlation between helpfulness_ratio and review_length is : {correlation2}")
print(f"The correlation in percentage is {correlation2:.2%}")
```

```
The correlation between helpfulness_ratio and review_length is : 0.039124168568013325
The correlation in percentage is 3.91%
```

Figure 4.9 : Correlations of helpfulness_ratio vs review_length

A correlation result of 0.0391 (or 3.91%) tells us that there is an even lower very weak relationship between the review length and its usefulness than the number of words in review and its usefulness. This result indicates that:

1. The length of the review does not significantly affect the usefulness
2. The flow of the text is more important than the number of characters
3. Other factors affecting the usefulness of reviews



```
# Visualization : Scatter plot helpfulness_ratio vs review_length
plt.figure(figsize=(10, 6))
sns.scatterplot(x=review_length, y=helpfulness_ratio, data=df_clear, color='blue')
plt.title('Helpfulness Ratio vs Review Length')
plt.xlabel('Review Length')
plt.ylabel('Helpfulness Ratio')
plt.show()
```

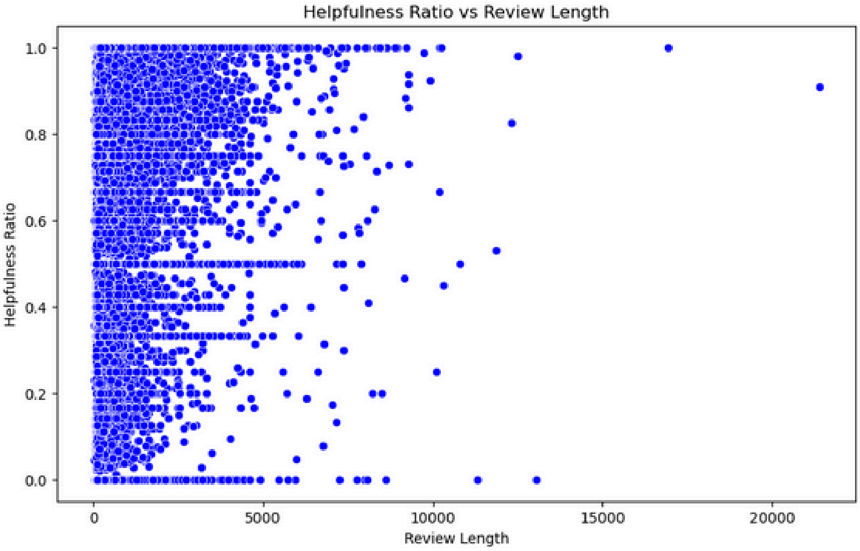


Figure 4.10 Scatter plot helpfulness_ratio vs review_length

The correlation coefficient (0.039 or 3.91%) between review_length and helpfulness_ratio indicates an extremely weak positive relationship. This provides insufficient evidence to reject the null hypothesis (H_0). The length of reviews in terms of character count does not significantly influence their perceived helpfulness. Other factors, beyond textual features, are likely more predictive of helpfulness metrics.

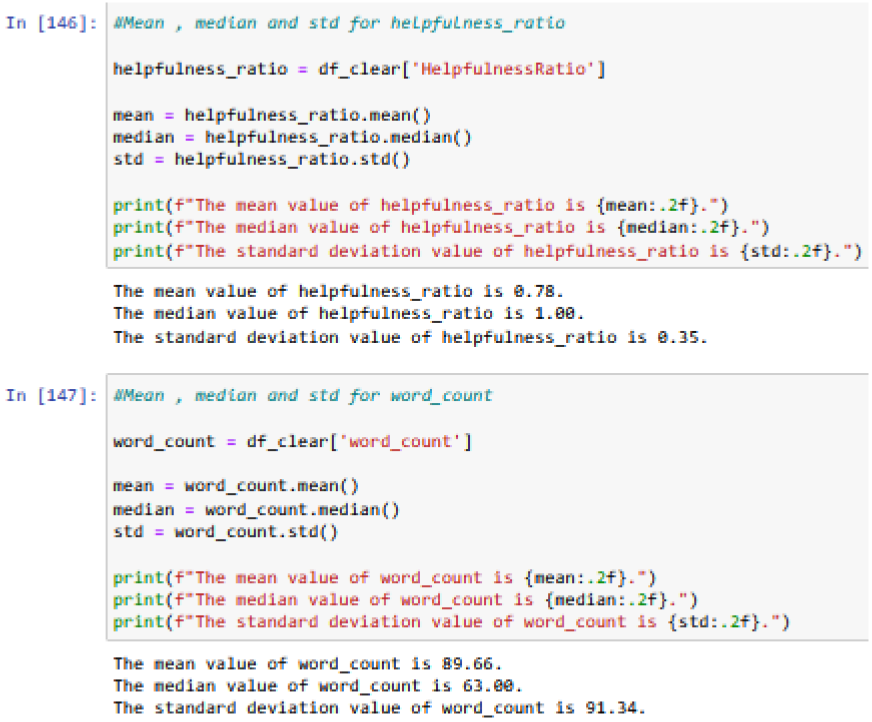
4.3 Statistical Analysis

Statistical analysis is the process of analyzing data to uncover patterns and make predictions from it.Statistical analysis provides a good basis for further predictions and accuracy. The figures obtained from calculations in the process of statistical analysis help us determine whether the given hypotheses are correct. The following statistical analysis stands out: T-Test and ANOVA Analysis.

4.3.1 Descriptive statistic

Descriptive statistics are brief informational coefficients that summarize a given dataset, which can be either a representation of the entire population or a sample of a population. Descriptive statistics are broken down into measures of central tendency and measures of variability (spread). Before dealing with statistical analysis, it is important to handle descriptive statistics and calculate the mean,median and standard deviation of certain columns that will be used for further calculations.

4.3.2 Mean,median and standard deviation calculations

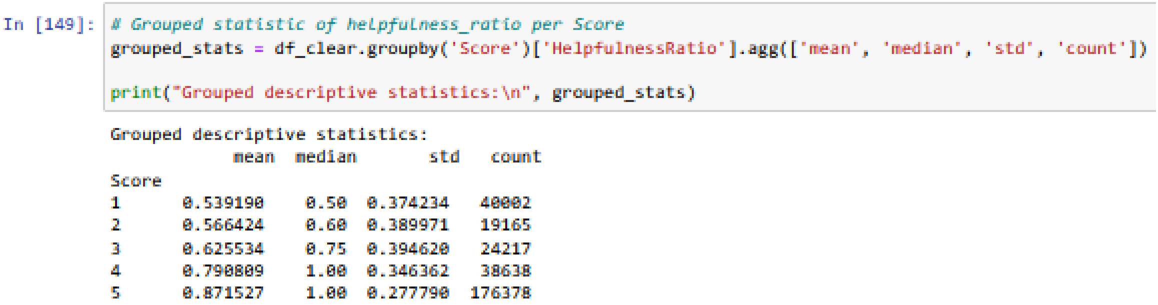


```
In [146]: #Mean , median and std for helpfulness_ratio

          helpfulness_ratio = df_clear['HelpfulnessRatio']

          mean = helpfulness_ratio.mean()
          median = helpfulness_ratio.median()
          std = helpfulness_ratio.std()

          print(f"The mean value of helpfulness_ratio is {mean:.2f}.")
          print(f"The median value of helpfulness_ratio is {median:.2f}.")
          print(f"The standard deviation value of helpfulness_ratio is {std:.2f}.")

          The mean value of helpfulness_ratio is 0.78.
          The median value of helpfulness_ratio is 1.00.
          The standard deviation value of helpfulness_ratio is 0.35.
```

```
In [147]: #Mean , median and std for word_count

          word_count = df_clear['word_count']

          mean = word_count.mean()
          median = word_count.median()
          std = word_count.std()

          print(f"The mean value of word_count is {mean:.2f}.")
          print(f"The median value of word_count is {median:.2f}.")
          print(f"The standard deviation value of word_count is {std:.2f}.")

          The mean value of word_count is 89.66.
          The median value of word_count is 63.00.
          The standard deviation value of word_count is 91.34.
```

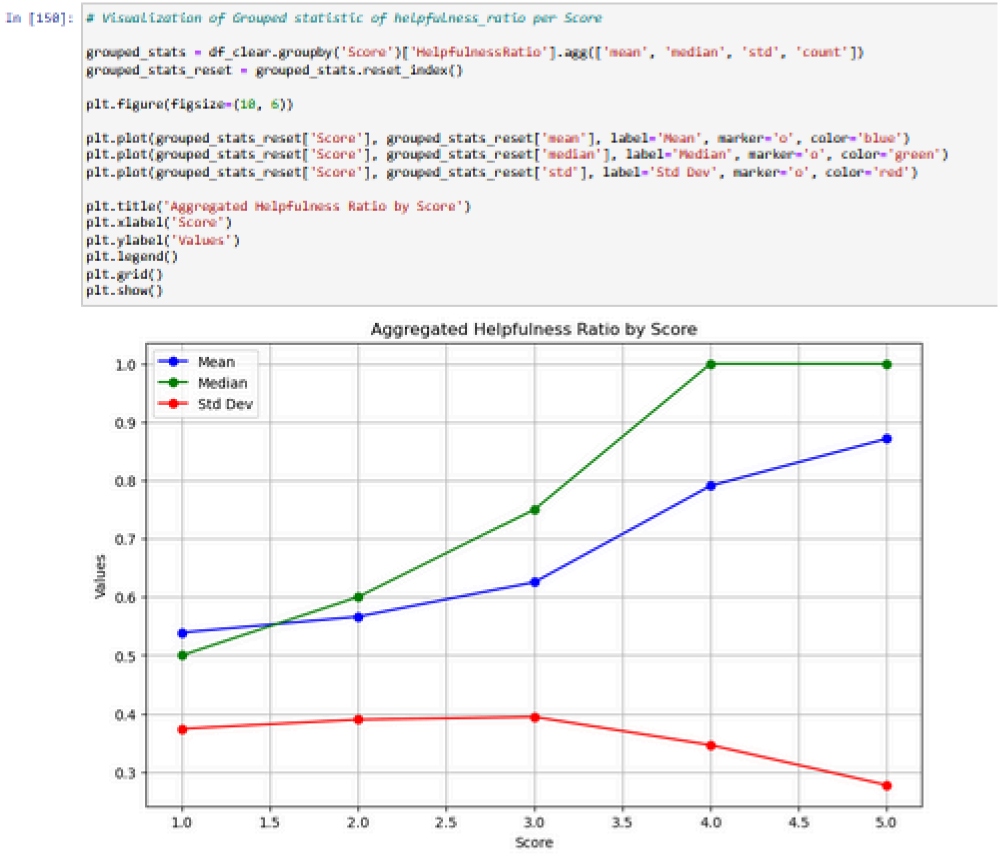
Figure 4.10 Mean,median and standard deviation calculations

Just as reviews can be grouped according to score, statistical values can also be grouped according to score.



```
In [149]: # Grouped statistic of helpfulness_ratio per Score
          grouped_stats = df_clear.groupby('Score')['HelpfulnessRatio'].agg(['mean', 'median', 'std', 'count'])

          print("Grouped descriptive statistics:\n", grouped_stats)

          Grouped descriptive statistics:
                     mean  median       std   count
          Score
          1      0.539190    0.50  0.374234   40002
          2      0.566424    0.60  0.389971   19165
          3      0.625534    0.75  0.394620   24217
          4      0.790809    1.00  0.346362   38638
          5      0.871527    1.00  0.277790  176378
```

Figure 4.11 Grouped statistical calculations

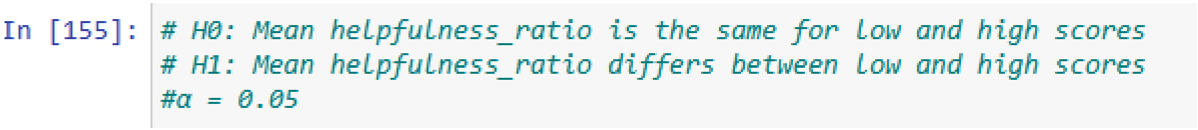


```
In [150]: # Visualization of Grouped statistic of helpfulness_ratio per Score

          grouped_stats = df_clear.groupby('Score')['HelpfulnessRatio'].agg(['mean', 'median', 'std', 'count'])
          grouped_stats_reset = grouped_stats.reset_index()

          plt.figure(figsize=(10, 6))

          plt.plot(grouped_stats_reset['Score'], grouped_stats_reset['mean'], label='Mean', marker='o', color='blue')
          plt.plot(grouped_stats_reset['Score'], grouped_stats_reset['median'], label='Median', marker='o', color='green')
          plt.plot(grouped_stats_reset['Score'], grouped_stats_reset['std'], label='Std Dev', marker='o', color='red')

          plt.title('Aggregated Helpfulness Ratio by Score')
          plt.xlabel('Score')
          plt.ylabel('Values')
          plt.legend()
          plt.grid()
          plt.show()
```

Figure 4.12 Visualization of grouped statistical calculations

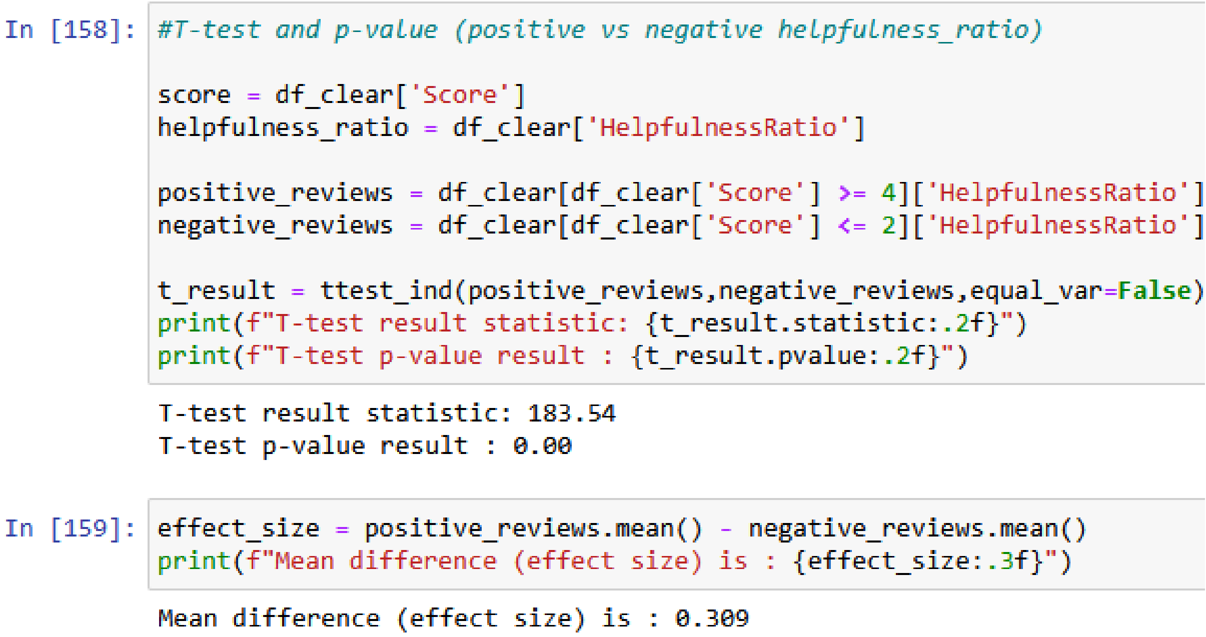
4.3.3 T-Test (Hypothesis Testing)

A t-test is a statistical test used to compare the means of two groups to determine if they are significantly different from each other. It is commonly used in hypothesis testing, where the null hypothesis states that there is no difference between the group means, while the alternative hypothesis suggests that there is a difference.



```
In [155]: # H0: Mean helpfulness_ratio is the same for low and high scores
          # H1: Mean helpfulness_ratio differs between low and high scores
          #α = 0.05
```

Figure 4.13 Setting hypothesis for T-Test



```
In [158]: #T-test and p-value (positive vs negative helpfulness_ratio)

          score = df_clear['Score']
          helpfulness_ratio = df_clear['HelpfulnessRatio']

          positive_reviews = df_clear[df_clear['Score'] >= 4]['HelpfulnessRatio']
          negative_reviews = df_clear[df_clear['Score'] <= 2]['HelpfulnessRatio']

          t_result = ttest_ind(positive_reviews,negative_reviews,equal_var=False)
          print(f"T-test result statistic: {t_result.statistic:.2f}")
          print(f"T-test p-value result : {t_result.pvalue:.2f}")

          T-test result statistic: 183.54
          T-test p-value result : 0.00

In [159]: effect_size = positive_reviews.mean() - negative_reviews.mean()
          print(f"Mean difference (effect size) is : {effect_size:.3f}")

          Mean difference (effect size) is : 0.309
```

Figure 4.14 T-Test calculations

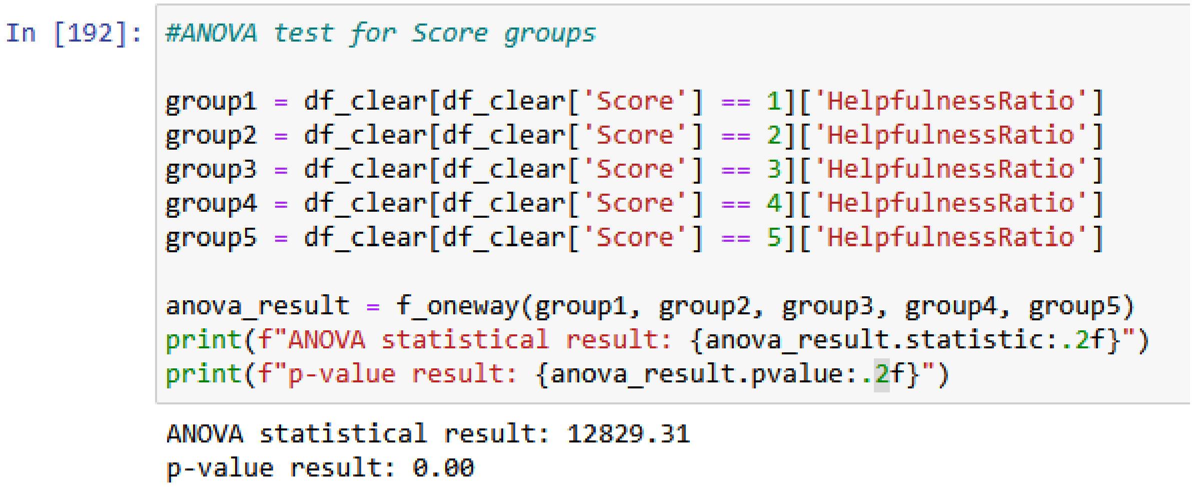
Since the T-test p-value result is 0.00 and it’s less than alpha (p-value < α), from the results above we can conclude that there is significant statistical difference between two groups of reviews and by Hypothesis testing we can reject H0 hypothesis and accept H1.This results means :

1. Positive reviews are significantly more helpful
2. Positive reviews receive higher helpfulness evaluations from users.
3. Negative reviews are less helpful

4.3.4 ANOVA Analysis

Analysis of variance (ANOVA) is a family of statistical methods used to compare the means of two or more groups by analyzing variance. Specifically, ANOVA compares the amount of variation between the group means to the amount of variation within each group. In this research, with ANOVA test would handle next hypothesis :

1. Null hypothesis (H_0): All groups have the same averages
2. Alternative hypothesis (H_1): At least one group has a different average



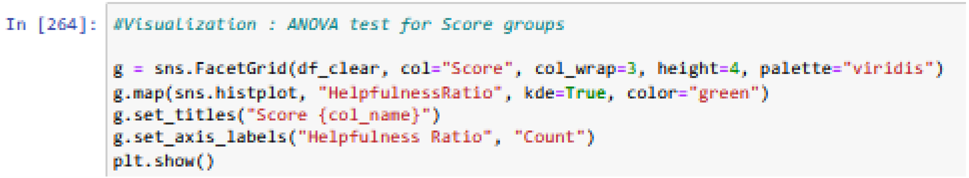
```
In [192]: #ANOVA test for Score groups

          group1 = df_clear[df_clear['Score'] == 1]['HelpfulnessRatio']
          group2 = df_clear[df_clear['Score'] == 2]['HelpfulnessRatio']
          group3 = df_clear[df_clear['Score'] == 3]['HelpfulnessRatio']
          group4 = df_clear[df_clear['Score'] == 4]['HelpfulnessRatio']
          group5 = df_clear[df_clear['Score'] == 5]['HelpfulnessRatio']

          anova_result = f_oneway(group1, group2, group3, group4, group5)
          print(f"ANOVA statistical result: {anova_result.statistic:.2f}")
          print(f"p-value result: {anova_result.pvalue:.2f}")

          ANOVA statistical result: 12829.31
          p-value result: 0.00
```

Figure 4.15 ANOVA Test calculations

The reviews are divided into groups based on their rating. The results show that there is very strong statistical evidence that the average Helpfulness Ratio differs between groups. The probability that these differences would be due to chance is less than 0.01%.An extremely high F-value can indicate two things:

1. Very large differences between groups
2. Very small variations within the groups themselves

Since the p-value is approximately 0.01, we can conclude that H_0 is rejected, which means that Alternative hypothesis (H_1) is accepted.



```
In [264]: #Visualization : ANOVA test for Score groups

          g = sns.FacetGrid(df_clear, col="Score", col_wrap=3, height=4, palette="viridis")
          g.map(sns.histplot, "HelpfulnessRatio", kde=True, color="green")
          g.set_titles("Score {col_name}")
          g.set_axis_labels("Helpfulness Ratio", "Count")
          plt.show()
```

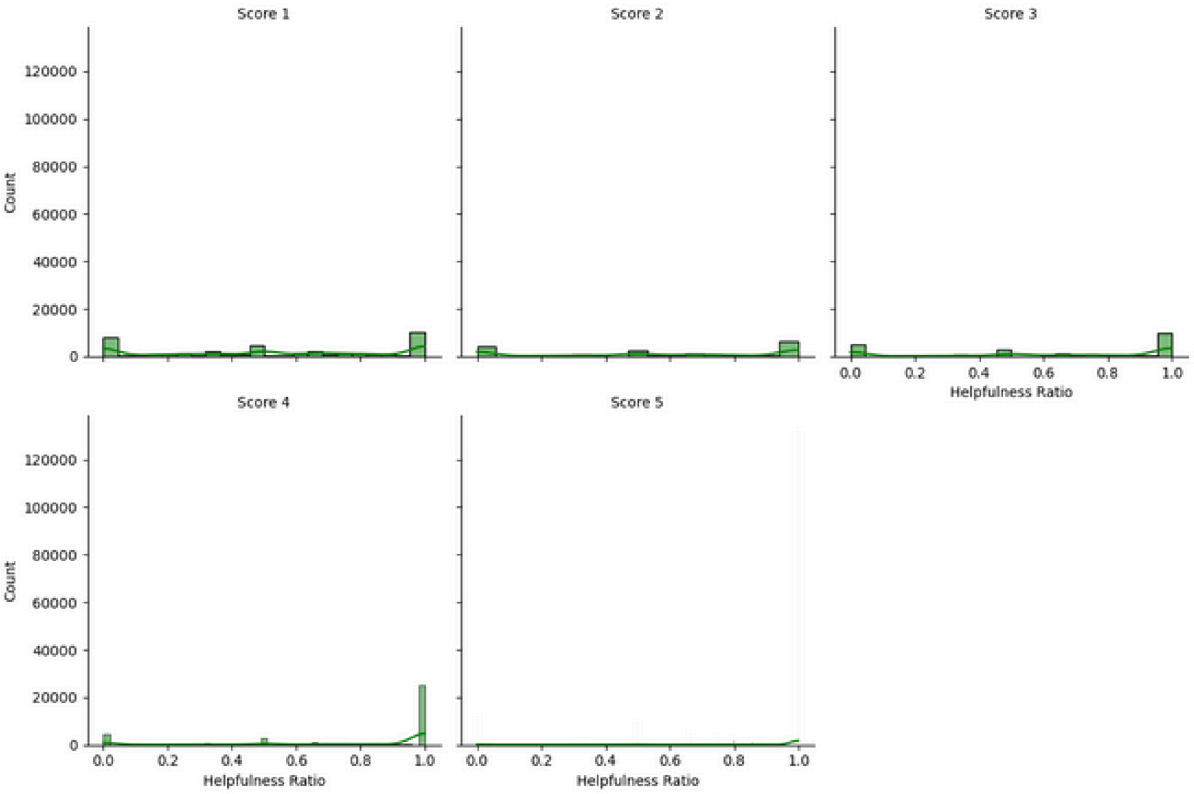
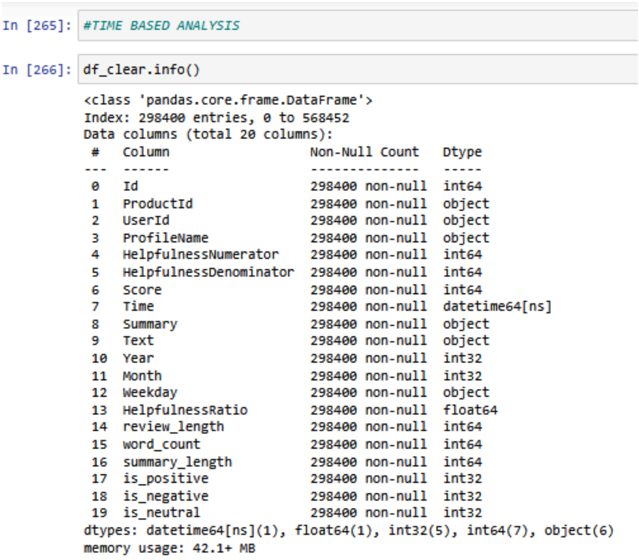


Figure 4.16 Visualization of ANOVA Test calculations

The helpfulness ratio significantly varies among score groups, with higher scores indicating higher helpfulness perceptions.

4.4 Temporal Analysis

Temporal analysis is the study of data as it changes over time.Temporal analysis will show us how user trends have changed over time, and how certain time periods have affected the number of reviews.Before any further time analyses, it is necessary to review the dataset in order to determine the columns that will be used for the time analysis.



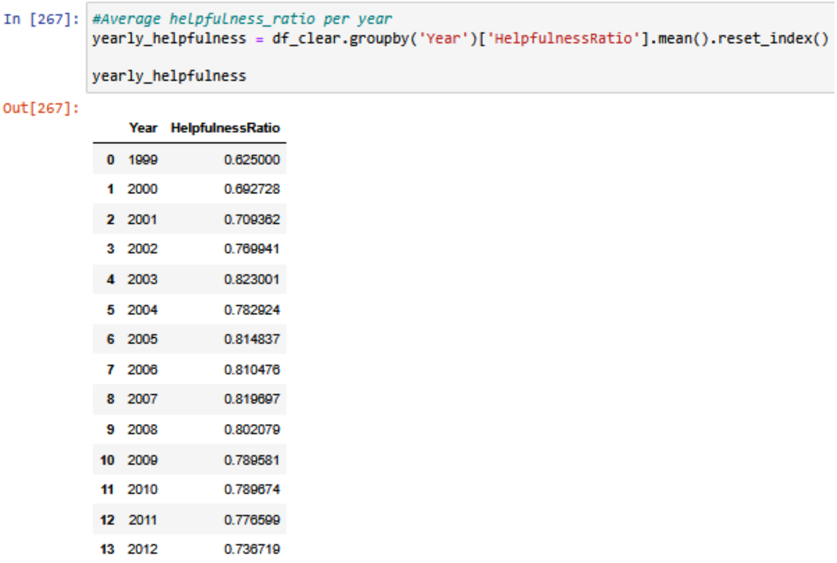
```
In [265]: #TIME BASED ANALYSIS

In [266]: df_clear.info()

          <class 'pandas.core.frame.DataFrame'>
          Index: 298400 entries, 0 to 568452
          Data columns (total 20 columns):
           #   Column                  Non-Null Count   Dtype
          ---  ------                  --------------   -----
           0   Id                      298400 non-null  int64
           1   ProductId               298400 non-null  object
           2   UserId                  298400 non-null  object
           3   ProfileName             298400 non-null  object
           4   HelpfulnessNumerator    298400 non-null  int64
           5   HelpfulnessDenominator  298400 non-null  int64
           6   Score                   298400 non-null  int64
           7   Time                    298400 non-null  datetime64[ns]
           8   Summary                 298400 non-null  object
           9   Text                    298400 non-null  object
           10  Year                    298400 non-null  int32
           11  Month                   298400 non-null  int32
           12  Weekday                 298400 non-null  object
           13  HelpfulnessRatio        298400 non-null  float64
           14  review_length           298400 non-null  int64
           15  word_count              298400 non-null  int64
           16  summary_length          298400 non-null  int64
           17  is_positive             298400 non-null  int32
           18  is_negative             298400 non-null  int32
           19  is_neutral              298400 non-null  int32
          dtypes: datetime64[ns](1), float64(1), int32(5), int64(7), object(6)
          memory usage: 42.1+ MB
```

Figure 4.17 Dataset Inspection

4.4.1 Yearly and Monthly Trends

Temporal analysis provides insights into evolving user behavior and platform activity over time. As this part of the research deals with, it is defined for two categories: Average helpfulness ratio by year and Number of reviews by month and year.

4.4.2 Average helpfulness ratio by year



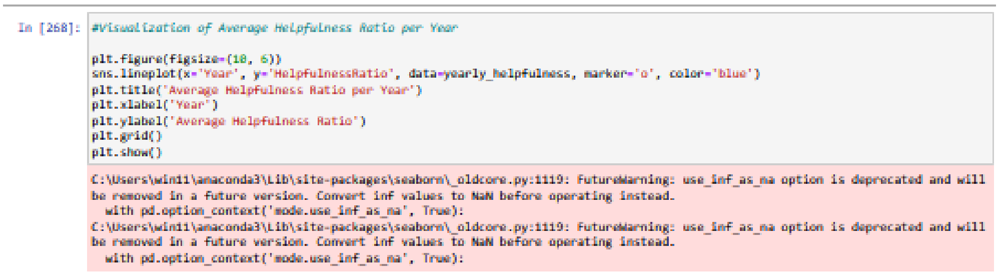
```
In [267]: #Average helpfulness_ratio per year
          yearly_helpfulness = df_clear.groupby('Year')['HelpfulnessRatio'].mean().reset_index()

          yearly_helpfulness
Out[267]:
```

	Year	HelpfulnessRatio
0	1999	0.625000
1	2000	0.692728
2	2001	0.709362
3	2002	0.769941
4	2003	0.823001
5	2004	0.782924
6	2005	0.814837
7	2006	0.810476
8	2007	0.819697
9	2008	0.802079
10	2009	0.789581
11	2010	0.789674
12	2011	0.776599
13	2012	0.736719

Figure 4.18 Average helpfulness ratio by year - DataFrame

DataFrame from above picture show us that the Helpfulness Ratio generally increases from 1999 to 2012 with initial value (1999): 0.625 (62.5%) and final value (2012): 0.7367 (73.67%) which lead to the total growth: ~11.17% points.Growth phases:

1. 1999-2003: Steady growth (0.625 → 0.823)
2. Biggest jump between 2002 and 2003 (+0.053)
3. 2003: Highest value in the series (0.823001 = 82.3%)
4. 2004-2012: Fluctuating trend with slight decline
5. Decline from peak in 2003 to end of 2012.



```
In [268]: #Visualization of Average Helpfulness Ratio per Year

          plt.figure(figsize=(10, 6))
          sns.lineplot(x='Year', y='HelpfulnessRatio', data=yearly_helpfulness, marker='o', color='blue')
          plt.title('Average Helpfulness Ratio per Year')
          plt.xlabel('Year')
          plt.ylabel('Average Helpfulness Ratio')
          plt.grid()
          plt.show()
```

```
C:\Users\win11\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will
be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\win11\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will
be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

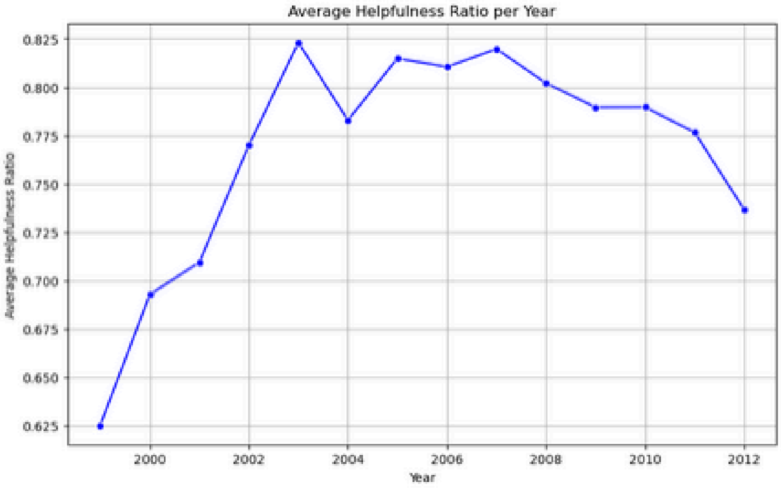
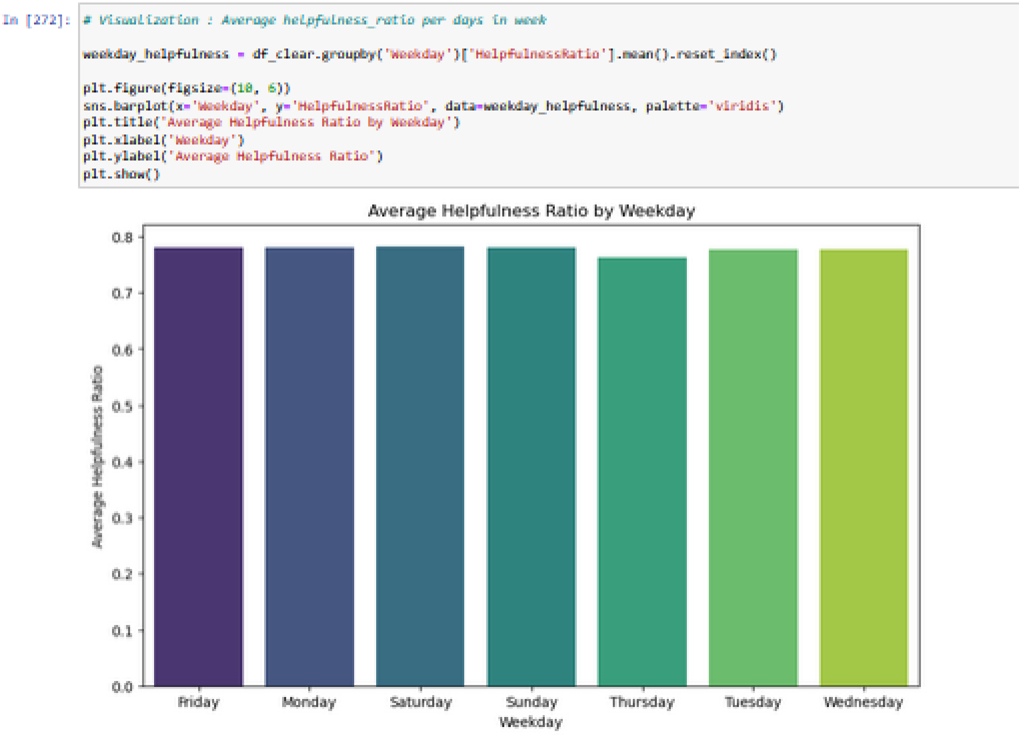


Figure 4.19 Visualization : Average helpfulness ratio by year

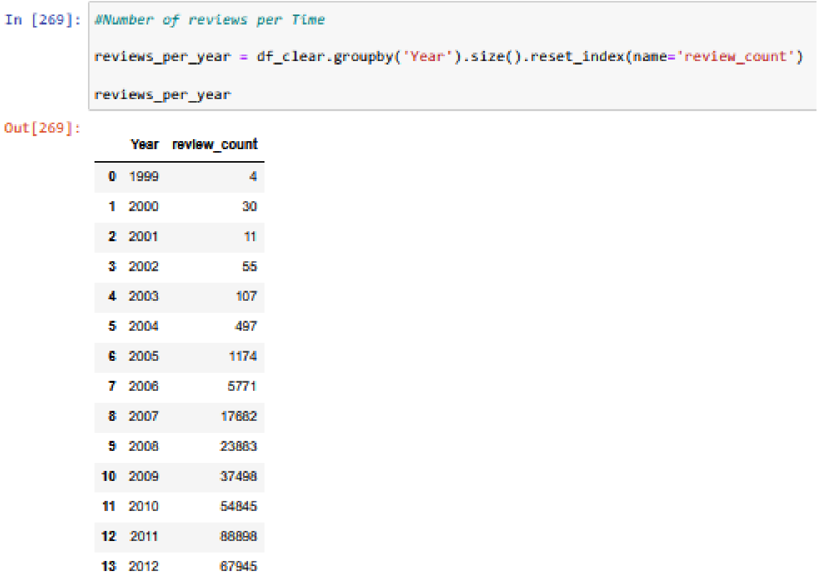


```
In [272]: # Visualization : Average helpfulness_ratio per days in week

          weekday_helpfulness = df_clear.groupby('Weekday')['HelpfulnessRatio'].mean().reset_index()

          plt.figure(figsize=(10, 6))
          sns.barplot(x='Weekday', y='HelpfulnessRatio', data=weekday_helpfulness, palette='viridis')
          plt.title('Average Helpfulness Ratio by Weekday')
          plt.xlabel('Weekday')
          plt.ylabel('Average Helpfulness Ratio')
          plt.show()
```

Figure 4.19 Visualization : Average helpfulness ratio by weekdays

Helpfulness ratios have remained stable over time, with minor fluctuations caused by external factors.

4.4.3 Number of Reviews by Month and Year



```
In [269]: #Number of reviews per Time

          reviews_per_year = df_clear.groupby('Year').size().reset_index(name='review_count')

          reviews_per_year
```

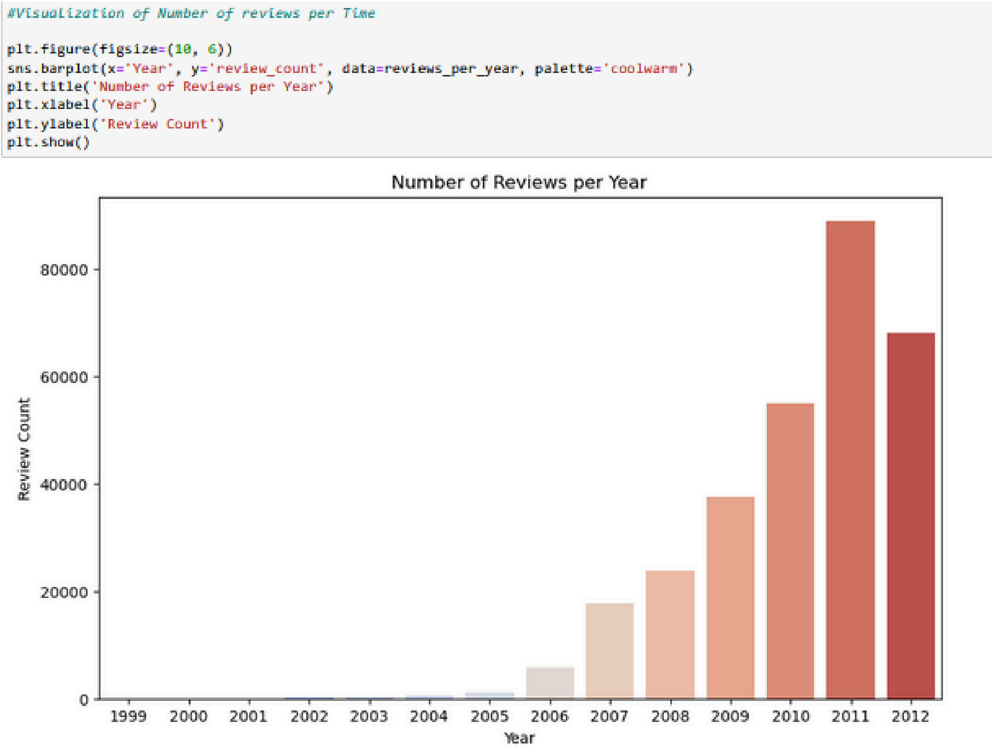
Out[269]:

	Year	review_count
0	1999	4
1	2000	30
2	2001	11
3	2002	55
4	2003	107
5	2004	497
6	2005	1174
7	2006	5771
8	2007	17682
9	2008	23863
10	2009	37498
11	2010	54845
12	2011	88898
13	2012	67945

Figure 4.20 Number of reviews per Year

DataFrame from the above picture shows us that the number of reviews per year has explosive growth. That growth can be defined in three stages :

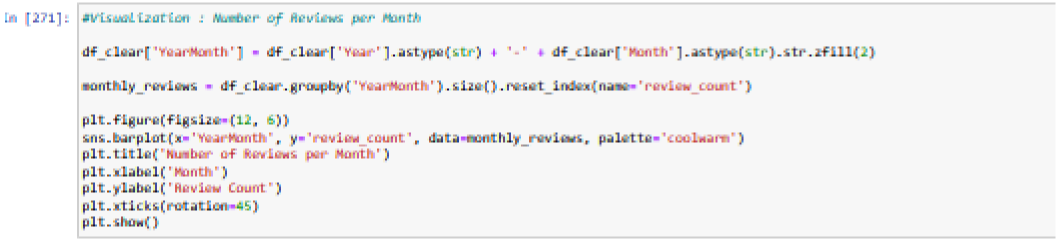
1. 1999-2005: Minimal number of reviews (4 to 1,174)
2. Since 2006: Explosive growth - more than 100x increase in a few years
3. 2011: Peak - 88,888 reviews



```
#Visualization of Number of reviews per Time

plt.figure(figsize=(10, 6))
sns.barplot(x='Year', y='review_count', data=reviews_per_year, palette='coolwarm')
plt.title('Number of Reviews per Year')
plt.xlabel('Year')
plt.ylabel('Review Count')
plt.show()
```

Figure 4.21 Number of Reviews per Year

The growth of reviews started in 2005, while it experienced an explosion in 2011, which was followed by a slight decline.Chen et al. (2008) highlighted the role of review timing and visibility in accumulating helpfulness votes. Consistent with their findings, this study shows seasonal peaks in review activity and user-perceived helpfulness. Temporal patterns suggest that increased user engagement during peak shopping seasons could amplify the perceived utility of reviews.



```
In [271]: #Visualization : Number of Reviews per Month

          df_clear['YearMonth'] = df_clear['Year'].astype(str) + '-' + df_clear['Month'].astype(str).str.zfill(2)

          monthly_reviews = df_clear.groupby('YearMonth').size().reset_index(name='review_count')

          plt.figure(figsize=(12, 6))
          sns.barplot(x='YearMonth', y='review_count', data=monthly_reviews, palette='coolwarm')
          plt.title('Number of Reviews per Month')
          plt.xlabel('Month')
          plt.ylabel('Review Count')
          plt.xticks(rotation=45)
          plt.show()
```

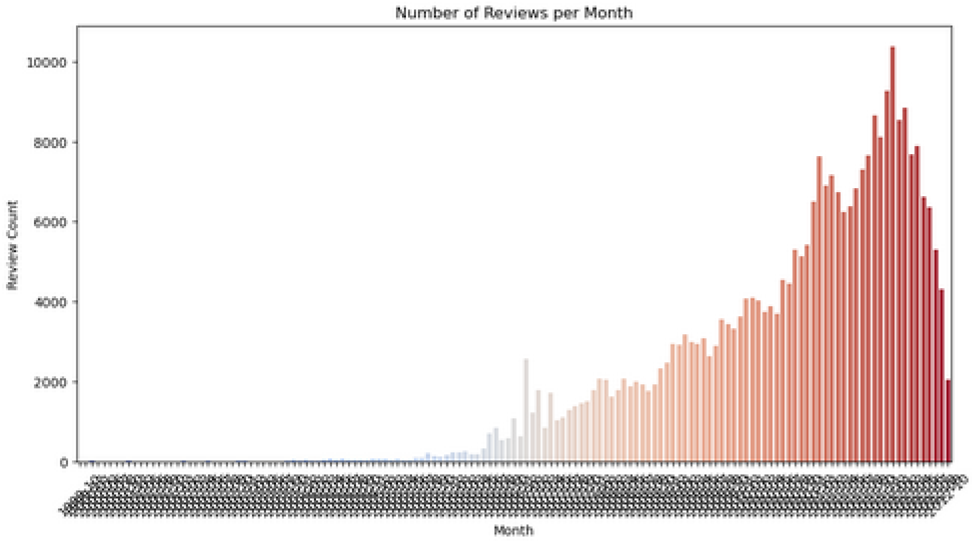
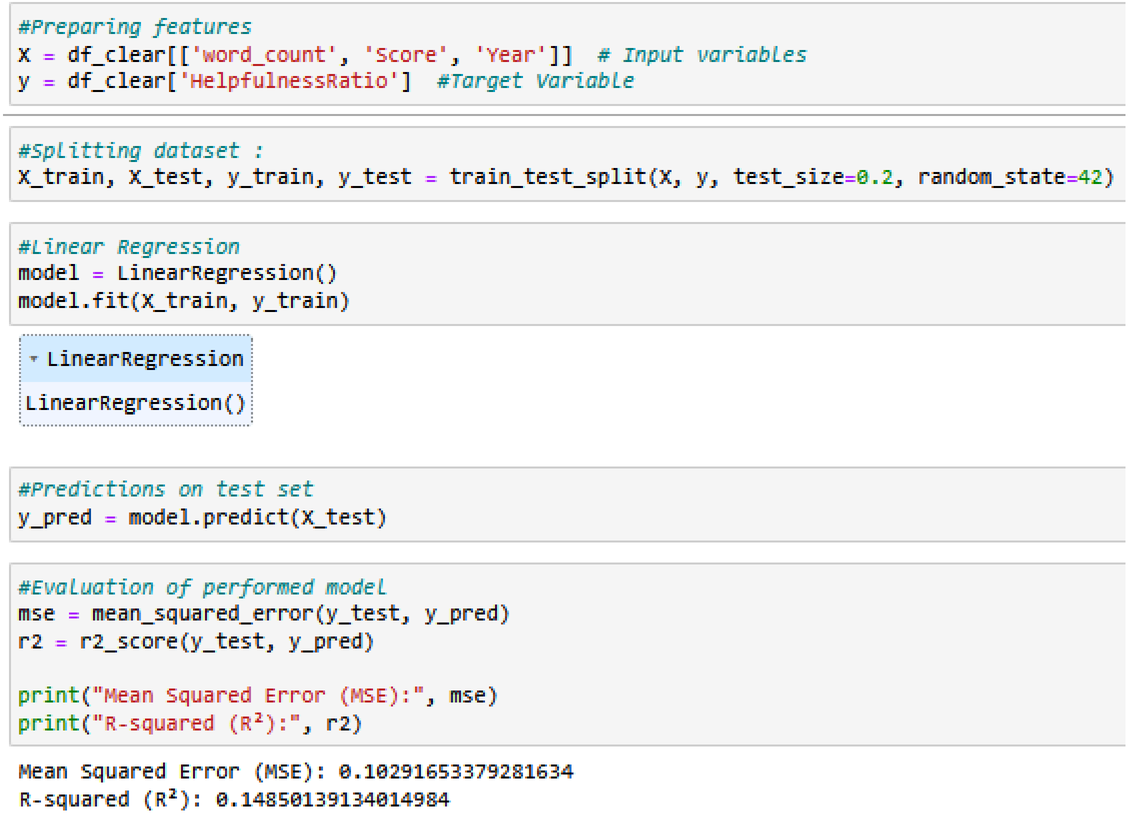


Figure 4.21 Number of Reviews per Month

The growth of reviews started in June 2005, while it experienced an explosion in December 2011. Review volumes have steadily increased over the observed time frame.

4.5 Predictive Modeling Results

To analyze factors influencing review helpfulness., as well as helpfulness of reviews that are already posted, Linear Regression Analysis was implemented. The objective of analysis is to examine how factors like word_count, Score, and Year impact HelpfulnessRatio . For the result of Linear Regression are used next matrices : Mean Squared Error (MSE) and R-squared (R2).



```
#Preparing features
X = df_clear[['word_count', 'Score', 'Year']]  # Input variables
y = df_clear['HelpfulnessRatio']  #Target Variable
```

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

```
#Linear Regression
model = LinearRegression()
model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

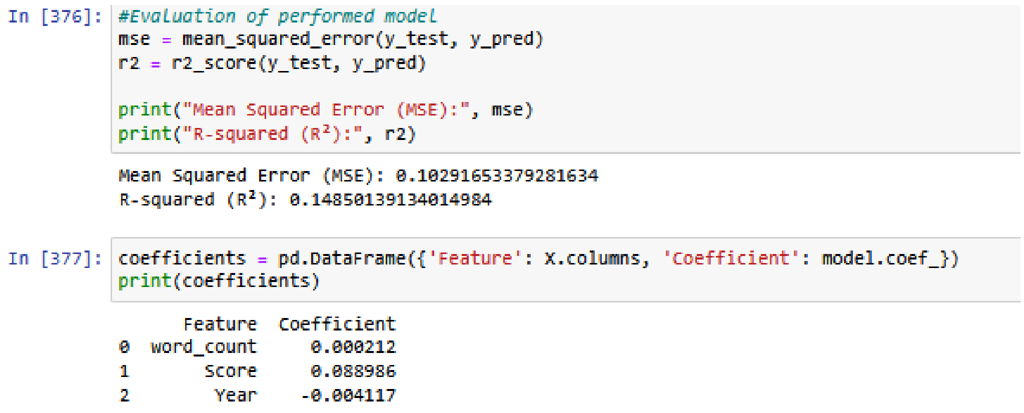
```
#Predictions on test set
y_pred = model.predict(X_test)
```

```
#Evaluation of performed model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error (MSE):", mse)
print("R-squared (R²):", r2)
```

```
Mean Squared Error (MSE): 0.10291653379281634
R-squared (R²): 0.14850139134014984
```

Figure 4.22 Linear Regression and its results

After selecting the columns that have an impact on the helpfulness ratio into a smaller dataset, the next part of Linear Regression is to divide that newly created dataset into a train-test followed by the results of the metrics obtained using linear regression.



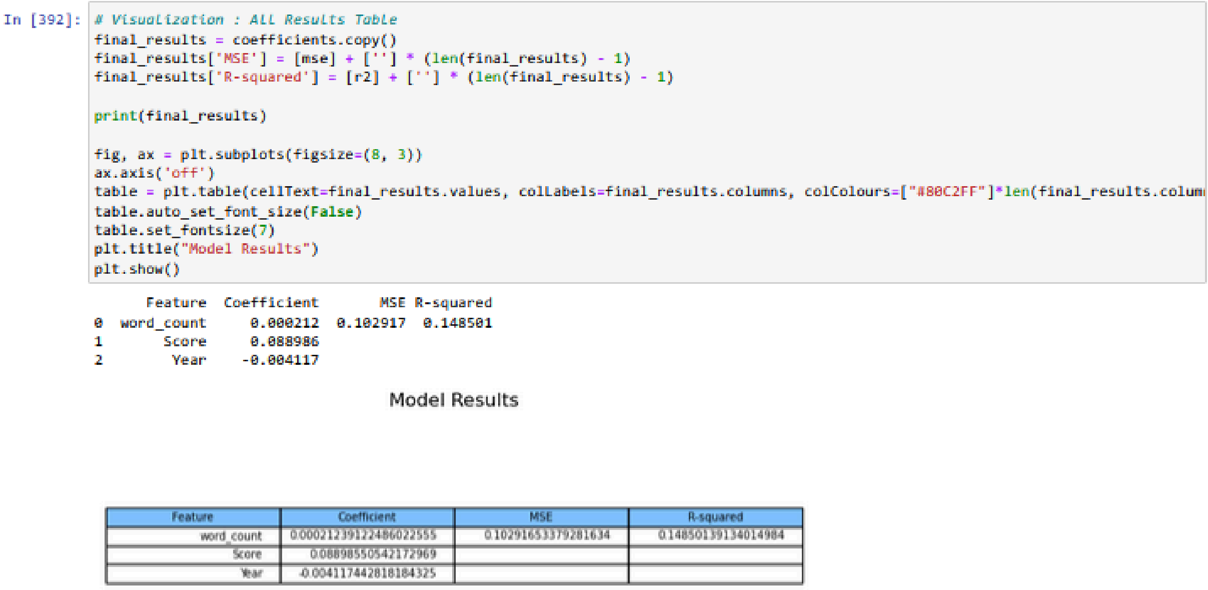
```
In [376]: #Evaluation of performed model
          mse = mean_squared_error(y_test, y_pred)
          r2 = r2_score(y_test, y_pred)

          print("Mean Squared Error (MSE):", mse)
          print("R-squared (R²):", r2)

          Mean Squared Error (MSE): 0.10291653379281634
          R-squared (R²): 0.14850139134014984

In [377]: coefficients = pd.DataFrame({'Feature': X.columns, 'Coefficient': model.coef_})
          print(coefficients)

                Feature  Coefficient
          0  word_count     0.000212
          1       Score     0.088986
          2        Year    -0.004117
```

Figure 4.23 Linear Regression results and coefficients

The model has low predictive power, explaining only 14.85% of the variance in the target variable. The average prediction error is approximately 0.32 on a 0-1 scale.

Feature Analysis (Coefficients):

1. Score (+0.089): The strongest positive predictor. Higher product ratings are associated with higher helpfulness scores.
2. Year (-0.004): A weak negative predictor. Older reviews tend to have slightly higher helpfulness ratios.
3. Word Count (+0.0002): A negligible positive effect. Review length has almost no impact on perceived helpfulness.



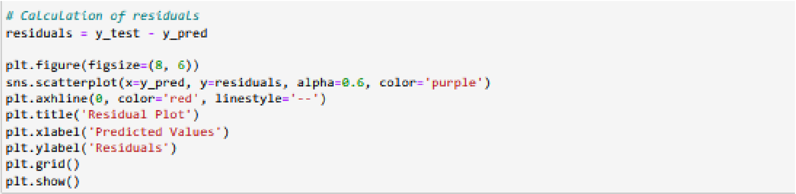
```
In [392]: # Visualization : All Results Table
final_results = coefficients.copy()
final_results['MSE'] = [mse] + [''] * (len(final_results) - 1)
final_results['R-squared'] = [r2] + [''] * (len(final_results) - 1)

print(final_results)

fig, ax = plt.subplots(figsize=(8, 3))
ax.axis('off')
table = plt.table(cellText=final_results.values, colLabels=final_results.columns, colColours=["#80C2FF"]*len(final_results.colum
table.auto_set_font_size(False)
table.set_fontsize(7)
plt.title("Model Results")
plt.show()
```

```
      Feature  Coefficient       MSE R-squared
0  word_count     0.000212  0.102917  0.148501
1       Score     0.088986                    
2        Year    -0.004117                    
```

Model Results

Feature	Coefficient	MSE	R-squared
word_count	0.00021239122486022555	0.10291653379281634	0.14850139134014984
Score	0.08898550542172969		
Year	-0.004117442818184325		

Figure 4.24 Table of all results



```
# Calculation of residuals
residuals = y_test - y_pred

plt.figure(figsize=(8, 6))
sns.scatterplot(x=y_pred, y=residuals, alpha=0.6, color='purple')
plt.axhline(0, color='red', linestyle='--')
plt.title('Residual Plot')
plt.xlabel('Predicted Values')
plt.ylabel('Residuals')
plt.grid()
plt.show()
```

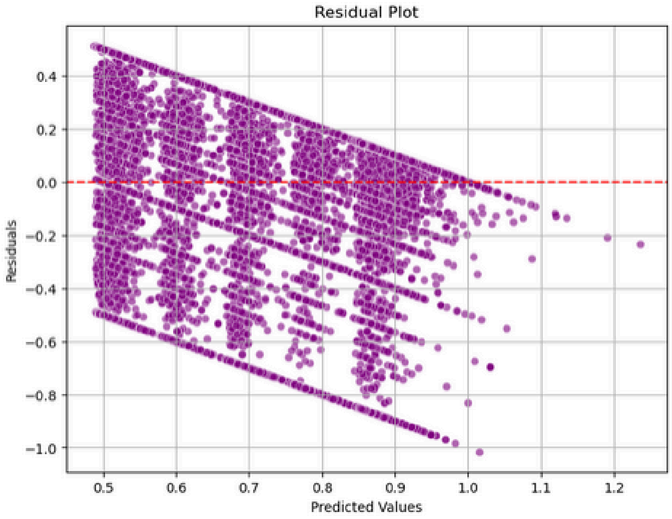


Figure 4.25 Predicted vs actual values

Low R2 indicates external factors not included in the model have a larger influence.Despite statistical significance, the model explains only a small portion of the variance in helpfulness ratios. Adding semantic content analysis or user behavior metrics might improve predictions.

4.6 Discussion

The findings presented in the Results section highlight several critical insights related to the helpfulness metrics of Amazon product reviews. This section discusses the implications of these results in the context of user behavior, review dynamics, and data-driven predictions.

4.6.1. The Impact of Review Scores

The analysis of review scores (Score) and their correlation with helpfulness demonstrates that higher scores are significantly associated with higher helpfulness ratios. Reviews rated 5 stars had the highest average helpfulness ratio (0.871), suggesting a positive bias in user perception of helpfulness for highly rated reviews. Conversely, reviews with lower scores (1-2 stars) exhibited significantly lower helpfulness ratios (mean between 0.539 and 0.566).

This bias towards positive reviews highlights the influence of sentiment on helpfulness metrics and suggests that users generally favor positive reviews when making purchasing decisions. Recognizing this trend is crucial for platforms seeking to balance visibility for critical (negative) reviews with helpful insights from positive reviews.

4.6.2. Textual and Numerical Features Correlation

Weak correlations between the HelpfulnessRatio and textual features (word_count and review_length) suggest that textual attributes alone are insufficient predictors of helpfulness. The highest correlation observed was only 4.1% for word_count, indicating minimal direct influence from review length or verbal complexity.

This finding emphasizes the importance of richer semantic analysis (e.g., sentiment mining or keyword extraction) in understanding what drives perceived helpfulness. Current textual metrics reveal surface-level information but do not capture deeper insights into content relevance.

4.6.3. Temporal Dynamics

Temporal analysis revealed a consistent average helpfulness ratio over time, with only minor fluctuations. Most reviews were concentrated in 2011 and 2012, representing 56.85% of the dataset. Additionally, monthly review activity peaked towards the end of the year, reflecting seasonal patterns in shopping and review behavior.

Understanding these temporal trends allows platforms and marketers to optimize content recommendations, addressing periods of high review activity and prioritizing impactful reviews during peak seasons.

4.6.4. Prediction Metrics

The predictive modeling results, using linear regression, show that elements like word_count, Score, and Year weakly influence HelpfulnessRatio. Although statistical significance was achieved, the model's low R2value (0.1485) indicates limited explanatory power. This suggests that additional semantic and behavioral features are required for stronger predictive performance (Liu et al., 2008; Filieri, 2015).Other factors, such as user engagement, product category, or sentiment analysis, likely have higher predictive utility.

This outcome underlines the need for more advanced approaches, such as machine learning classifiers or natural language processing (NLP) models, to capture nuanced patterns that traditional regression overlooks. Incorporating semantic features, product-specific factors, and user demographics can improve predictive performance.

4.6.5. Observations on Professional Reviews

The identification of "professional" reviewers (users with more than 5 reviews) helps categorize users based on activity levels. While this metric alone does not wholly define review utility, it introduces a valuable dimension for studying user behavior and identifying trusted voices in the feedback community.Encouraging engagement from professional reviewers, while maintaining unbiased visibility for casual reviews, may enhance the overall helpfulness of review datasets. Platforms could use these insights to refine algorithms that rank and recommend reviews.

5. Conclusion

This research systematically examined the helpfulness of online product reviews, focusing on Amazon food reviews as a case study. Through the analysis of 568,454 reviews spanning over a decade, we employed data science methodologies—including exploratory data analysis (EDA), statistical inference, and predictive modeling—to uncover patterns, relationships, and dynamics that influence the perceived usefulness of customer feedback. The findings highlight several critical insights and offer actionable contributions for improving review and recommendation systems on e-commerce platforms.

The study demonstrates that higher review scores show a clear and statistically significant association with higher helpfulness ratios, reflecting a positive bias in user perceptions. Reviews rated 5 stars achieved the highest average helpfulness ratio (0.871), whereas lower-rated reviews (1–2 stars) exhibited significantly lower levels of perceived utility. This trend indicates that sentiment—as expressed through ratings—holds considerable weight in shaping the perceived helpfulness of reviews. For review systems and vendors, this insight underscores the importance of balancing visibility between critical (negative) and positive reviews to ensure informed decision-making for consumers.

Additionally, while textual features such as word_count and review_length are often considered indicators of review depth, their weak positive correlations with helpfulness ratios (4.1% and 3.9%, respectively) suggest that quantitative textual attributes are insufficient predictors of perceived usefulness. Semantic aspects and contextual relevance, which may require advanced techniques such as natural language processing (NLP), can provide deeper insights into what makes a review valuable to users. Encouraging meaningful and concise feedback could enhance engagement and improve review utility.

Temporal trends emerged as another influential factor, highlighting seasonal patterns in review activity and helpfulness. Most reviews were concentrated in 2011 and 2012, representing 56.85% of the dataset, with high activity levels peaking in December—likely driven by holiday shopping behaviors. Such trends are valuable for platforms seeking to optimize engagement strategies, such as prioritizing impactful reviews during periods of heightened customer activity.

The predictive modeling conducted through linear regression provided limited explanatory power ((R^2 = 0.1485)), highlighting the complexity of helpfulness metrics and the influence of factors not captured in the dataset. While review score emerged as the strongest positive predictor of helpfulness, other variables—including word count and temporal attributes—exhibited negligible predictive weight. These results emphasize the necessity for incorporating richer features—such as sentiment, semantic content, and user credibility—into future models to enhance predictive accuracy and uncover nuanced patterns.

The identification of "professional" reviewers (users with more than five reviews) categorizing user behavior and potential indicators of credibility within the review ecosystem. Although professional reviewers tend to exhibit distinct engagement patterns, platforms must balance their visibility with casual reviewers to ensure unbiased representation and diversity in customer feedback.

While this study provides valuable insights, it is not without limitations. The analysis relied primarily on structured attributes, leaving out richer textual and behavioral data that might significantly influence helpfulness metrics. Furthermore, predictive modeling focused on interpretability rather than complexity, restricting the scope of insights to linear relationships.While this study focuses on structured features, further research can build upon findings by Ghose and Ipeirotis (2011) and Filieri

(2015), incorporating sentiment analysis and contextual word embeddings Future work can address these gaps by incorporating machine learning algorithms, sentiment analysis, and broader datasets encompassing multiple product categories and e-commerce platforms.

In conclusion, this study underscores the value of data-driven approaches in analyzing user-generated content and improving review systems on e-commerce platforms. By identifying key factors that influence helpfulness, the findings offer practical implications for enhancing user experience, guiding informed purchasing decisions, and optimizing engagement strategies. As e-commerce continues to expand, integrating advanced analytics and methodologies will provide deeper insights into user behavior, fostering trust and satisfaction in online marketplaces.

6. References

Chen, P.-Y., Dhanasobhon, S., & Smith, M. D. (2008). *All reviews are not created equal: The disaggregate impact of reviews on sales on Amazon.com*. Social Science Research Network (SSRN). https://doi.org/10.2139/ssrn.918083

Forman, C., Ghose, A., & Wiesenfeld, B. (2008). *Examining the relationship between reviews and sales: The role of reviewer identity disclosure in electronic markets*. Information Systems Research, 19(3), 291–313. https://doi.org/10.1287/isre.1080.0193

Ghose, A., & Ipeirotis, P. G. (2011). *Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics*. IEEE Transactions on Knowledge and Data Engineering, 23(10), 1498–1512. https://doi.org/10.1109/TKDE.2010.188

Liu, J., Cao, Y., Lin, C.-Y., Huang, Y., & Zhou, M. (2007). *Low-quality product review detection in opinion summarization*. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL) (pp. 334–342). Association for Computational Linguistics. https://aclanthology.org/D07-1035/

Mudambi, S. M., & Schuff, D. (2010). *What makes a helpful online review? A study of customer reviews on Amazon.com*. MIS Quarterly, 34(1), 185–200. https://doi.org/10.2307/20721420

Schlosser, A. E. (2011). *Can including pros and cons increase the helpfulness and persuasiveness of online reviews? The interactive effects of ratings and arguments*. Journal of Consumer Psychology, 21(3), 226–239.https://doi.org/10.1016/j.jcps.2011.04.002

Pan, Y., & Zhang, J. Q. (2011). *Born unequal: A study of the helpfulness of user-generated product reviews*. Journal of Retailing, 87(4), 598–612.https://doi.org/10.1016/j.jretai.2011.05.002

Filieri, R. (2015). *What makes online reviews helpful? A diagnosticity–adoption framework*. Journal of Business Research, 68(6), 1261–1270. https://doi.org/10.1016/j.jbusres.2014.11.006

Liu, Y., Huang, X., An, A., & Yu, X. (2008). *Modeling and predicting the helpfulness of online reviews*. Proceedings of the 8th IEEE International Conference on Data Mining (ICDM), 443–452 https://doi.org/10.1109/ICDM.2008.94

Racherla, P., & Friske, W. (2012). *Perceived "usefulness" of online consumer reviews: An exploratory investigation across three services categories*. Electronic Commerce Research and Applications, 11(6), 548–559. https://doi.org/10.1016/j.elerap.2012.06.003