SENTIMENT ANALYSIS OF CUSTOMER REVIEW DATA

Post Graduate Program in Data Science Engineering

Location: Hyderabad **Batch: PGPDSE-FT Jul23**

Submitted by

Bandi Nikhil
Kata Ganesh
Jupally Rishivedh
Palepu Vaishnavi
Pamarti Vibha
C. Manideep Reddy

Mentored by Mr. Manvendra Singh

Table Of Content

ABSTRACT

Sentiment analysis helps to determine the hidden intentions of the respective authors on each topic and provides an evaluation report on the polarity of each document. Polarity can be positive, negative, or neutral. It can be seen that most of the data related to sentiment analysis includes feedback provided by different customers on any product. Therefore, the reviews can be correctly classified into any type of classification based on polarity to gain sufficient knowledge about the product. In this project, we propose an approach to classify datasets created based on sentiment analysis into different polar groups. The results obtained concerning the accuracy values of the algorithm using different performance parameters applied to the dataset are critically examined.

The purpose of this project is to perform sentiment analysis of product-based reviews by using data. This project uses online product reviews collected from “amazon.com”. We expect to do Classification at the review level of survey data with promising results.

INTRODUCTION

Sentiment analysis, also known as opinion mining, analyses people's opinions and feelings about entities such as products, organizations, and their attributes. There are many social media on the web where you can easily get information about any product through reviews, blogs, and comments. These comments not only help analysts study the reaction to any commodity or product but may also allow customers to learn about demand in any given region. Various machine learning techniques are often used by researchers and professionals to perform an analysis of people's emotions optimally. Sentiment analysis examines how the text conveys an emotion. Customer feedback, survey responses, and product reviews are all frequently used. This is useful in a variety of situations, including social media monitoring, reputation management, and customer service.

For example, analyzing thousands of product reviews can provide important feedback on product pricing and features. People's willingness to interact with a company and their overall perception of a brand are heavily influenced by public opinion. According to Podium research, 93% of shoppers believe that online reviews influence their purchasing decisions. After reading some negative reviews, users may not want to give the company a chance. They don't check if the feedback is genuine or not. They will choose another way. In this situation, companies that carefully monitor their reputation can quickly address issues and improve their operations based on feedback. In the age of information, such an analysis allows accurate measurement of people's attitudes towards companies.

Two main categories of machine learning techniques are mainly used in sentiment analysis: supervised learning and unsupervised learning.

In supervised learning, datasets are labeled during analysis and trained to provide meaningful outputs that are useful for decision-making. Unlike supervised learning, unsupervised learning does not provide labeled data, which is very difficult to process. Different clustering algorithms are often investigated to solve this problem. This project highlights the effect of supervised learning techniques on labeled data.

A natural language processing (NLP) technique is used to determine whether data is positive, negative, or neutral. It is often used with contextual data to help organizations track brand and product sentiment in consumer feedback and better understand customer desires. This tool helps businesses extract information from unstructured text found on the Internet, including emails, blog posts, support tickets, web chats, social media channels, forums, and comments.

Types of Sentiment Analysis

● Step-by-step sentiment analysis If polarity accuracy is important to your business, expand your polarity category to include multiple positive and negative levels. very positive Positive neutral negative very negative This is commonly referred to as step-by-step or detailed sentiment analysis and can be used to interpret 5-star ratings in reviews.

For example:

Very positive = 5 stars

Very negative = 1 star

● Recognizing emotions Emotion Analysis Emotion Recognition allows you to recognize emotions such as happiness, frustration, anger, and sadness across polarities. Many emotion recognition systems use vocabularies (ie, lists of words and the emotions they convey) or complex machine learning algorithms. One of the disadvantages of using a dictionary is that people express their feelings in different ways. Some words that usually express anger, such as bad and kill (e.g. your product is so bad or your customer support is killing me) can also express happiness (e.g. this is terrible or You are killing it).

● Aspect-based sentiment analysis Typically, when analyzing textual sentiment, you want to know which aspects or specific features people refer to positively, neutrally, or negatively. This is where aspect-based sentiment analysis comes in handy.

For example, in this product review, if you say, "This camera's battery life is too short," a facet-based classifier can determine that the sentence expresses a negative opinion about battery life. of the desired product

● Multilingual sentiment analysis Multilingual sentiment analysis can be difficult. It requires a lot of preprocessing and resources. Most of these resources are available online (eg, emotion dictionaries), but others need to be developed (eg, translated corpora and noise detection algorithms). However, you need to know how to code to use them. Alternatively, you can use the language classifier to automatically detect the language in text and train a custom sentiment analysis model to classify the text into the language of your choice.

Overview

1.1. E-Commerce Industry Review- Current Practices in Sentiment Analysis of Customer Reviews

The intense competition to attract and maintain customers online is compelling businesses to implement novel strategies to enhance customer experiences. It is becoming necessary for companies to examine customer reviews on online platforms such as Amazon to understand better how customers rate their products and services. The purpose of this study is to investigate how companies can conduct sentiment analysis based on Amazon reviews to gain more insights into customer experiences. The dataset selected for this capstone consists of customer reviews and ratings from consumer reviews of Amazon products. Amazon product reviews enable a business to gain insights into customer experiences regarding specific products and services. The study will enable companies to pinpoint the reasons for positive and negative customer reviews and implement effective strategies to address them accordingly.

- Customer Feedback Aggregation:

E-commerce platforms collect and aggregate customer reviews for products. This data includes textual feedback, ratings, and other relevant information.

- Product Ranking and Recommendations:

Sentiment analysis is used to influence product rankings and recommendations. Positive reviews contribute to higher rankings, making products more visible to potential buyers.

- Quality Assurance and Product Improvement:

Companies analyze sentiments to identify product strengths and weaknesses. Patterns in negative reviews can highlight areas of improvement.

- Competitor Analysis:

E-commerce platforms often conduct sentiment analysis not only on their reviews but also on competitor reviews. This helps in benchmarking and understanding market sentiments.

- Fraud Detection and Review Authenticity:

Sentiment analysis is used to detect fraudulent reviews or those that may not be authentic. Unusual statement patterns may signal potential issues.

- Marketing and Advertising Strategies:

It also contributes to the development of marketing strategies. Positive sentiments in reviews can be incorporated into advertising campaigns.

- Dynamic Pricing Strategies:

It is sometimes used to gauge market sentiments about pricing. E-commerce platforms may adjust prices based on customer sentiments and competitor pricing.

In summary, sentiment analysis in the e-commerce industry is a multifaceted tool used for product improvement, customer service enhancement, marketing strategies, and overall business intelligence. As technology advances, the industry is likely to continue refining and expanding its use of sentiment analysis to better understand and respond to customer sentiments.

1.2. Literature Survey

a. Publications:

- “Mining and Summarizing Customer Reviews “(2004) by Minqing Hu and Bing Liu:

This sentimental work introduced a lexicon-based approach to sentiment analysis. The researchers proposed a method to mine product features and opinions from customer reviews. They utilized sentiment lexicons to determine the sentiment orientation of reviews and presented techniques for summarizing customer opinions.

- “Thums Up? Sentiment Classification using Machine Learning Techniques” (2002) by Bo Pang and Lillian Lee:

Pang and Lee’s work was instrumental in popularizing the application of machine learning, particularly Support Vector

Machine (SVM), for sentiment classification. They demonstrated the effectiveness of supervised learning in determining the sentiment of movie reviews, showcasing the potential of statistical methods in sentiment analysis.

These publications represent pivotal moments in the development of sentiment analysis methodologies. Researchers have built upon these foundations to explore advanced techniques, such as deep learning models, aspect-based analysis, and more contributing to the evolving landscape of sentiment analysis in customer reviews.

b. Applications:

- Aspect-Based Sentiment Analysis in Reviews:

Researchers have explored aspect-based sentiment analysis, going beyond overall sentiment to identify sentiments towards specific aspects or features mentioned in reviews.

- Multimodal Sentiment Analysis:

With the rise of visual content, there is ongoing research on combining text-based sentiment analysis with image or video analysis. This is particularly relevant in e-commerce where users may provide visual feedback along with textual reviews.

- Cross-Domain Sentiment Analysis

Researchers have explored techniques for adapting sentiment analysis models trained on one domain to perform well on reviews from a different domain. This is crucial for models to generalize effectively across various types of products or services.

c. Past Research

- Title: "Customer Reviews and Business Intelligence: A Linguistic Approach"

o Authors: Gautam Pant et al.

o Year: 2017

o Key Findings: This study explored the linguistic aspects of customer reviews on Amazon. It delved into the relationship between language use and business intelligence, emphasizing the role of reviews in understanding customer behavior.

- Title: "Fake Reviews: An Overview"

o Authors: Muntasir Raihan Rahman et al.

o Year: 2019

o Key Findings: Addressing the issue of fake reviews, this research surveyed methods for detecting and mitigating fraudulent reviews. It highlighted the importance of maintaining the integrity of sentiment analysis outcomes.

d. Ongoing Research

▪ Title: "Cross-Domain Sentiment Analysis for Amazon Products"

o Focus: Exploring the challenges and opportunities in adapting sentiment analysis models trained on one domain to handle diverse product categories on Amazon.

▪ Title: "Ethical Considerations in Sentiment Analysis of Amazon Customer Reviews"

o Focus: Examining ethical considerations in the collection, analysis, and use of sentiment data from Amazon reviews, addressing issues such as user privacy and bias.

▪ Title: "Impact of Sentiments on Amazon Sales: A Longitudinal Study"

o Focus: Investigating the relationship between sentiments expressed in customer reviews and the sales performance of products on the Amazon platform.

• These ongoing research directions highlight the evolving nature of sentiment analysis in the context of Amazon reviews. Researchers continue to explore advanced methodologies, ethical considerations, and the dynamic interplay between sentiments and business outcomes on the e-commerce platform.

ABOUT DOMAIN

Retail analytics focuses on providing insights related to sales, inventory, customers, and other important aspects crucial for merchants' decision-making process. The discipline encompasses several granular fields to create a broad picture of a retail business' health, and sales alongside overall areas for improvement and reinforcement. Essentially, retail analytics is used to help make better choices, run businesses more efficiently, and deliver improved customer service analytics. Retail analysis goes beyond superficial data analysis, using techniques like data mining and data discovery to sanitize datasets to produce actionable BI insights that can be applied in the short term.

Moreover, companies use these analytics to create better snapshots of their target demographics. By harnessing sales data analysis, retailers can identify their ideal customers according to diverse categories such as age, preferences, buying patterns, location, and more. Essentially, the field is focused not just on parsing data, but also on defining what information is needed, how best to gather it, and most importantly, how it will be used. Electronic commerce is the buying and selling of goods or services through the internet. The products are advertised on e-commerce websites developed by online retailers. These companies have domain expertise in the products and services they offer to buyers. Various companies like Amazon, Big Bazaar, Flipkart, Snapdeal offer products that are essential to daily needs, from books to mobiles, electronic gadgets to clothing, groceries to kitchen essentials.

<u>ECommerce offers various advantages:</u>

● A comprehensive range of options for products concerning gender, age, brand, fashion, and price range.

- Round-the-clock availability to select and purchase various items of desire and need.
- Payment options from net banking, credit cards, e-wallet, cash on delivery, etc. Privacy of personal information and guarantee of secured transactions. Refund option in case of return of goods if the user changes his/her mind from purchase. Once registered on the portal
- Ease of login, view wish list, the status of the order placed, history of purchases, repeat order from past purchases, option to save credit card details for quick pay, the intimation of availability of a product in case it is not in stock, in case product is in the wishlist.
- Options to purchase items from the web as well as mobile platforms for the ecommerce web portal. Electronic commerce is buying and selling goods or services through the Internet. The products are advertised on e-commerce websites developed by online retailers.

  Types of eCommerce portals are B2B, B2C, C2C. Some of the features like an exhaustive range of options for product to select from, 24 x7 availability of product to select desired items, options of paying like net banking, credit cards, cash on delivery, a guarantee of secured transactions, refund option in case of return of purchased items, wishlist, order status, quick pay options.

PROBLEM STATEMENT

"In the landscape of electronic product reviews, the challenge persists in effectively employing sentiment analysis to derive actionable insights. Despite the prevalence of sentiment analysis tools, accurately gauging the sentiments expressed in electronic product reviews remains a complex task. The issue revolves around the need for a more nuanced and contextually aware sentiment analysis approach that can decipher the subtleties and intricacies of user sentiments towards electronic products. Addressing this challenge involves developing or employing sentiment analysis models tailored to the specific jargon, technical aspects, and domain-specific language used in electronic product reviews. The objective is to create a more reliable and accurate sentiment analysis framework capable of capturing the diverse spectrum of opinions and emotions expressed in these reviews. Enhancing sentiment analysis in the realm of electronic product reviews is crucial for empowering consumers with trustworthy insights and supporting manufacturers in understanding user sentiments to improve product development and customer satisfaction."

Dataset

DATA

a. Data Dictionary

Columns in the Dataset:

- brand: the brand or manufacturer of the product

(categorical: “Amazon”, “Amazon Fire”, “Amazon Echo brand”, “Amazon Fire TV”, “Amazon basics”)

- manufacturer: the company or entity that produced or manufactured the item

(categorical:” Amazon”, “Amazon basics”, “Amazon Digital Services, Inc”, “Amazon Digital Services”, “Amazon.com”)

- reviews.doRecommend: indicates if the reviewer recommends the product

(categorical)

- reviews. rating: the rating given in the review (e.g.: star rating) (numeric: float64)

- reviews. text: the text content of the review(categorical)

- review. title: the title or summary of the review

B. Variable Categorization (numeric and categorical):

- Number of numeric variables: 1
- Number of categorical variables: 5

c.Alternate Sources of Dataset

• When collecting data from alternate sources, it’s essential to ensure that you have the right to use and analyze the data for sentiment analysis. Additionally, consider the context and potential biases associated with each source to ensure a comprehensive understanding of customer sentiments.

• Access product reviews from other online platforms dedicated to reviews.

• Explore forums or discussion boards related to products on websites like Quora, Stack Exchange or dedicated forums for specific product categories.

• Explore reviews and ratings on other online retailers that sell Amazon products. Many products are available on multiple platforms, and each may have its own set of customer reviews.

• Alternate columns that can be added from sources are:

- Reviewer's Verified Purchase Status: Indicate whether the reviewer has verified their purchase on Amazon. Verified reviews may carry more weight as they confirm that the reviewer actually bought the product.
- Reviewer's Location: Include the geographical location of the reviewer. This information can be useful for understanding regional variations in sentiment or identifying localized issues.

• Adding these columns can enrich your dataset and allow for more nuanced analyses.

Data Processing

- Data Info

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35631 entries, 0 to 35630
Data columns (total 7 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Unnamed: 0          35631 non-null  int64  
 1   brand               35631 non-null  object 
 2   manufacturer        35631 non-null  object 
 3   reviews.doRecommend 35037 non-null  object 
 4   reviews.rating      35598 non-null  float64
 5   reviews.text        35630 non-null  object 
 6   reviews.title       35626 non-null  object 
dtypes: float64(1), int64(1), object(5)
memory usage: 1.9+ MB
```

- Pre-Processing Data

```
df.isnull().sum()
```

```
Unnamed: 0               0
brand                    0
manufacturer             0
reviews.doRecommend    594
reviews.rating          33
reviews.text             1
reviews.title            5
dtype: int64
```

We can observe that there are null values in the columns reviews.doRecommend,reviews.rating, reviews.text and reviews.title

```
df.dropna(subset=['reviews.doRecommend','reviews.rating','reviews.text','reviews.title'],inplace=True)
```

```
df.isnull().sum()
```

```
Unnamed: 0             0
brand                  0
manufacturer           0
reviews.doRecommend    0
reviews.rating         0
reviews.text           0
reviews.title          0
dtype: int64
```

We have also dropped the Unamed:0 column which was a duplicate column for serial no, which is a unique identifier

```
df.drop('Unnamed: 0',axis=1,inplace=True)
```

```
df
```

	brand	manufacturer	reviews.doRecommend	reviews.rating	reviews.text	reviews.title
0	Amazon	Amazon	True	5.0	This product so far has not disappointed. My c...	Kindle
1	Amazon	Amazon	True	5.0	great for beginner or experienced person. Boug...	very fast
2	Amazon	Amazon	True	5.0	Inexpensive tablet for him to use and learn on...	Beginner tablet for our 9 year old son.
3	Amazon	Amazon	True	4.0	I've had my Fire HD 8 two weeks now and I love...	Good!!!
4	Amazon	Amazon	True	5.0	I bought this for my grand daughter when she c...	Fantastic Tablet for kids
...	...	...	...	...	...	...
35626	Amazon	Amazon	True	3.0	Unable to download the Instagram app. which st...	The camera is awful!
35627	Amazon	Amazon	True	5.0	Pros: 5 Ghz wifi, supports expansion cardsCons...	Very good for reading, email, internet
35628	Amazon	Amazon	True	5.0	Bought it for my little grandson he likes game...	very good tablet
35629	Amazon	Amazon	True	5.0	I just like reading amd playing games on it..B...	Works great
35630	Amazon	Amazon	True	4.0	Allows for reading of my Kindle lib and even p...	Works as advertised

35033 rows × 6 columns

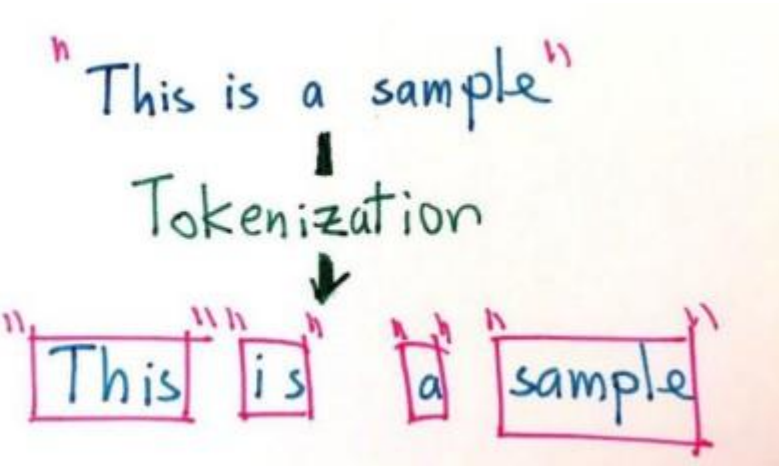
```python
import string
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer


def preprocess_text_adjusted(text):
    # Lowercasing
    text = text.lower()

    # Remove special characters and digits, punctuation
    text = ''.join(char for char in text if char.isalnum() or char.isspace() or char in ['!', '?', ','])

    # Tokenization
    words = word_tokenize(text)

    # Remove stopwords
    stop_words = set(stopwords.words('english'))
    words = [word for word in words if word not in stop_words]

    # Lemmatizing
    lemmatizer = WordNetLemmatizer()
    words = [lemmatizer.lemmatize(word) for word in words]
    # Join the processed words back into a sentence
    processed_text = ' '.join(words)

    return processed_text
df['reviews.text_adjusted'] = df['reviews.text'].apply(preprocess_text_adjusted)
print(df)
```

The above code is performing text preprocessing on the column reviews.text.

The function “preprocess_text_adjusted” takes a text input and performs the following steps:

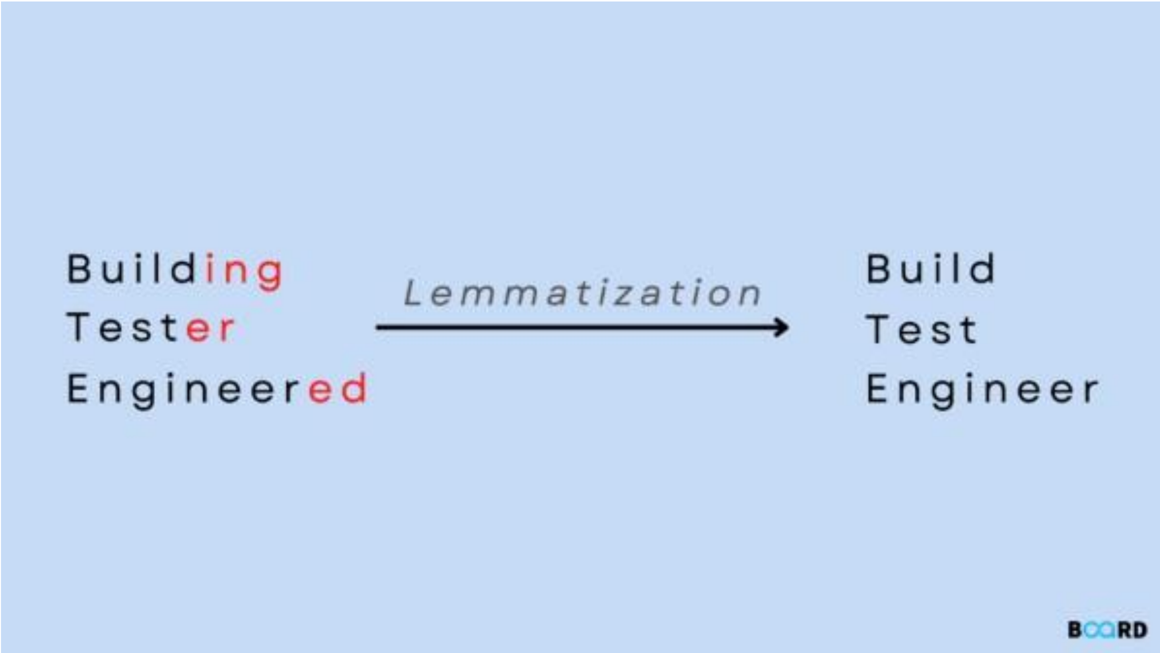
1. Lowercasing: Converts the entire text to lowercase to ensure uniformity.

2. Removing special characters and digits: Uses a list comprehension to keep on; alphanumeric characters, spaces, and specific punctuation marks.



3. Tokenization: Splits the text into a list of words using the “word_tokenize” function.

4. Removing stopwords: Removes common English stopwords using NLTK’s “stopwords” set.



5. Lemmatizing:Applies lemmatization to reduce words to their base or root form using the WordNet lemmatizer.

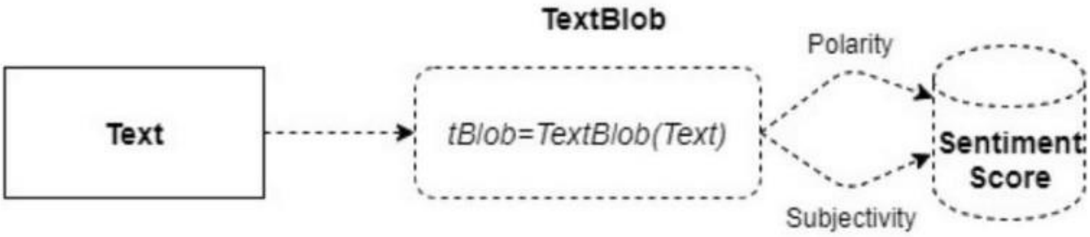
6. Joining words: Joins the processed words back into a sentence using “.join”. “preprocess_text_adjusted” applies to each row in the “review.text” column and creates a new column “review.text_adjusted”. This type of preprocessing is commonly used in Natural language processing (NLP) tasks to clean and standardize text data for analysis or machine learning models.

```
from textblob import TextBlob
df['sentiment'] = df['reviews.text_adjusted'].apply(lambda text: 'positive' if TextBlob(text).sentiment.polarity > 0.1
                                                    else ('negative' if TextBlob(text).sentiment.polarity < -0.1 else 'neutral'))
```

```
df['sentiment'].value_counts()
```

```
positive    29186
neutral      4063
negative     1784
```

The code snippet provides a quick and automated way to assess the sentiments of the preprocessed text data.



• Textblob for sentiment analysis: The code leverages TextBlob, a NLP library in Python, for sentiment analysis. TextBlob provides a simple API for common natural language processing tasks, including sentiment analysis.

• Sentiment Categorization: Sentiments are categorized into three classes: ”positive”, “negative”, and “neutral”.

• The sentiment categorization is based on the polarity score calculated by TextBlob.

• If the polarity score is greater than 0.1, the sentiment is categorized as “positive”.

• If the polarity score is less than -.01, the sentiment is categorized as “negative”.

• If the polarity score is between -0.1 and 0.1(inclusive) , the sentiment is categorized as “neutral”.

• The results of the sentiment analysis are store in a new column named “sentiment” within the data frame.

```python
from nltk import bigrams
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer

def get_bigrams(sentiment_category):
    selected_reviews = df[df['sentiment'] == sentiment_category]['reviews.text_adjusted']
    text = ' '.join(selected_reviews)

    # Tokenizing the text
    tokens = word_tokenize(text)

    # Removing stopwords
    stop_words = set(stopwords.words('english'))
    filtered_tokens = [token for token in tokens if token.lower() not in stop_words]

    # Lemmatization
    lemmatizer = WordNetLemmatizer()
    lemmatized_tokens = [lemmatizer.lemmatize(token) for token in filtered_tokens]

    # Removing non-alphanumeric characters
    lemmatized_tokens = [token for token in lemmatized_tokens if token.isalnum() or token.isspace()]

    # Removing single characters
    lemmatized_tokens = [token for token in lemmatized_tokens if len(token) > 1]

    # Extracting bigrams
    sentiment_bigrams = list(bigrams(lemmatized_tokens))

    return sentiment_bigrams

positive_bigrams = get_bigrams('positive')
negative_bigrams = get_bigrams('negative')
neutral_bigrams = get_bigrams('neutral')
print("Positive Review Bigrams:", positive_bigrams[:10])
print("Negative Review Bigrams:", negative_bigrams[:10])
print("Neutral Review Bigrams:", neutral_bigrams[:10])
```

```
Positive Review Bigrams: [('great', 'beginner'), ('beginner', 'experienced'), ('experienced', 'person'), ('person', 'bought'),
('bought', 'gift'), ('gift', 'love'), ('love', 'inexpensive'), ('inexpensive', 'tablet'), ('tablet', 'use'), ('use', 'learn')]
Negative Review Bigrams: [('really', 'like'), ('like', 'tablet'), ('tablet', 'would'), ('would', 'given'), ('given', 'star'),
('star', 'sometimes'), ('sometimes', 'push'), ('push', 'start'), ('start', 'several'), ('several', 'time')]
Neutral Review Bigrams: [('product', 'far'), ('far', 'disappointed'), ('disappointed', 'child'), ('child', 'love'), ('love', 'u
se'), ('use', 'like'), ('like', 'ability'), ('ability', 'monitor'), ('monitor', 'control'), ('control', 'content')]
```

• The above code defines a function `get_bigrams` that extracts bigrams (pairs of consecutive words) from a collection of reviews based on their sentiment category. It then applies this function to three sentiment categories ('positive', 'negative', 'neutral') and prints the first 10 bigrams for each category

• bigrams: This function is part of NLTK and is used to extract bigrams from a sequence of words.

• word_tokenize: Tokenizes words in a text.

• stopwords: Provides a set of common English stopwords.

• WordNetLemmatizer: Lemmatizes words using WordNet's lexical database.

• The `get_bigrams` function takes a sentiment category as an argument and extracts bigrams from reviews belonging to that sentiment category

• It first selects the reviews for the specified sentiment category from the DataFrame and concatenates them into a single text string.

• The text is tokenized into a list of words using `word_tokenize`. • Stopwords are removed from the list of tokens.

• Lemmatization is applied to reduce words to their base or root form using the WordNet lemmatizer.

• Non-alphanumeric characters are removed, and single characters are filtered out.

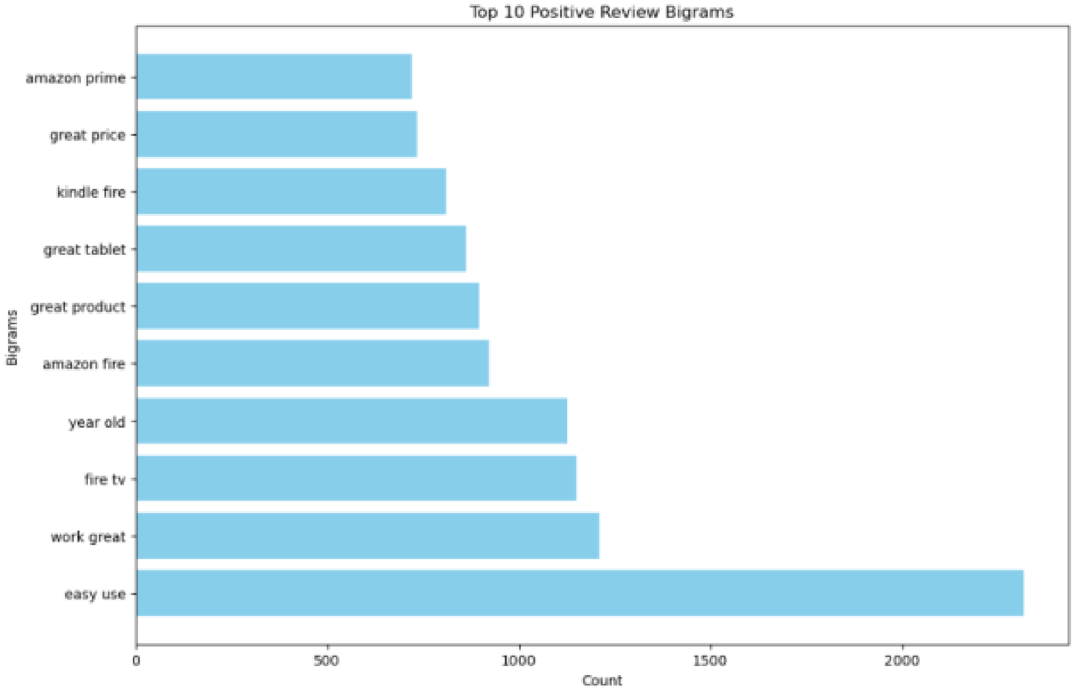
• The function returns a list of bigrams for the specified sentiment category.

• The first 10 bigrams for each sentiment category are printed.

• This kind of analysis can be useful for understanding the language patterns associated with different sentiments in the reviews.

```
from collections import Counter

# Count the occurrences of each bigram in each sentiment category
positive_bigram_counts = Counter(positive_bigrams)
negative_bigram_counts = Counter(negative_bigrams)
neutral_bigram_counts = Counter(neutral_bigrams)

# Get the top N bigrams for each sentiment category
num_top_bigrams = 10
top_positive_bigrams = positive_bigram_counts.most_common(num_top_bigrams)
top_negative_bigrams = negative_bigram_counts.most_common(num_top_bigrams)
top_neutral_bigrams = neutral_bigram_counts.most_common(num_top_bigrams)

# Function to plot bigrams
def plot_bigrams(top_bigrams, title):
    bigrams, counts = zip(*top_bigrams)
    bigrams = [' '.join(bigram) for bigram in bigrams]  # Join the bigrams into a single string
    plt.figure(figsize=(12, 8))
    plt.barh(bigrams, counts, color='skyblue')
    plt.title(title)
    plt.xlabel('Count')
    plt.ylabel('Bigrams')
    plt.show()

# Plot the top N bigrams for each sentiment category
plot_bigrams(top_positive_bigrams, 'Top 10 Positive Review Bigrams')
```

The above code segment analyzes the occurrence of bigrams in different sentiment categories(“positive”, ”negative”, ”neutral”) and visualizes the top N bigrams for each catgeory. The “Counter” class is imported from the “collections” module. It is used to count the occurrence of elements in a collection. The plot_bigrams function is defined to create bar plots for visualizaing the top bigrams. The resulting plots provide insights into the language pattern associated with different sentiments, helping to identify recurring word pairs in each sentiment catgeory



• The frequent occurrence of the bigram "easy use" suggests that customers who express positive sentiments often associate the ease of use with the product or service.

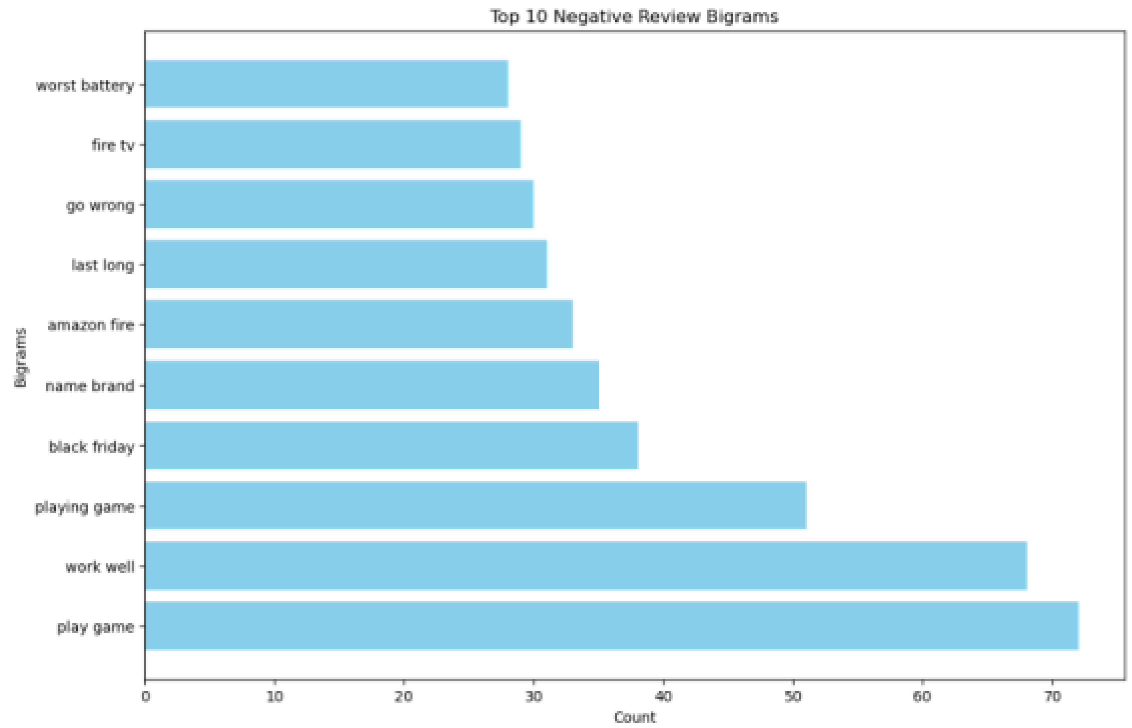
• The high count indicates that a significant number of customers find the product or service easy to use. This is a positive signal, as usability is a key factor in customer satisfaction.

• Businesses can leverage this insight in marketing materials, emphasizing the user-friendly nature of their offerings.

• Highlighting the "easy use" aspect in promotional campaigns, product descriptions, or user manuals can reinforce positive perceptions.

• If applicable, businesses can use this information to inform future product development efforts. Understanding what customers appreciate about the usability can guide improvements or enhancements.

• In summary, recognizing and acting on insights related to positive sentiments, such as ease of use, can have a positive impact on customer satisfaction and overall business success.



• The bigrams "play game" and "work well" are the top two most frequently occurring bigrams in negative reviews.

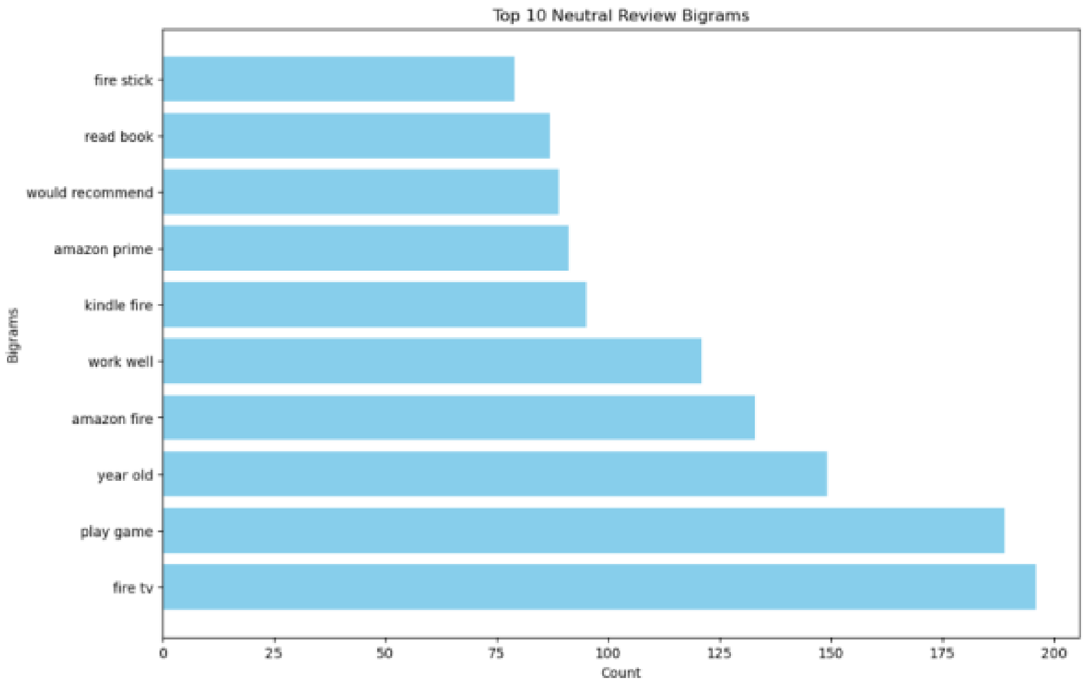
• The presence of "play game" as a common bigram in negative reviews suggests that customers may be expressing dissatisfaction or issues related to the product or service's performance in gaming scenarios.

• It could indicate concerns such as poor gaming experience, glitches, or performance issues that negatively impact the user's ability to play games.

• For the "play game" bigram, businesses should investigate and address any issues related to the product's performance in gaming scenarios. This could involve optimizing gaming features, addressing bugs, or improving the overall gaming experience. Understanding the context of phrases like "work well" in negative reviews is crucial. It might indicate that customers expected the product to work well but experienced issues, emphasizing the importance of aligning marketing claims with actual performance.

• Analyzing these insights can guide efforts to enhance user experience in specific scenarios (such as gaming) and improve overall product performance.

• In summary, identifying and addressing specific issues mentioned in negative reviews, can contribute to overall product improvement and customer satisfaction.



• Neutral sentiments often indicate a balanced view where customers may provide feedback without a strong inclination towards satisfaction or dissatisfaction.

• Bigrams in neutral reviews may represent aspects that customers find neither particularly positive nor negative.

• Analyzing neutral bigrams can highlight areas that may not be outstanding but are still noteworthy to customers.

• Businesses can use this information to identify potential areas for improvement or refinement in their products or services.

• Neutral bigrams might involve discussions about specific features, functionalities, or experiences that are neither exceptionally positive nor negative.

• Understanding these features can guide product development and marketing strategies.

• Neutral reviews may involve comparisons with other products or experiences. Analyzing these comparisons can provide insights into customer preferences and expectations.

• Consider product enhancements or modifications based on neutral feedback to move certain aspects from neutral to positive.

```python
from wordcloud import WordCloud
import matplotlib.pyplot as plt
def generate_wordcloud(sentiment, ax):
    # Combining reviews for the specified sentiment into a single text
    text = ' '.join(df[df['sentiment'] == sentiment]['reviews.text_adjusted'])

    # Generating the word cloud
    wordcloud = WordCloud(width=400, height=200, background_color='white').generate(text)

    # Ploting the word cloud
    ax.imshow(wordcloud, interpolation='bilinear')
    ax.axis('off')  # Turn off axis labels
    ax.set_title(f'{sentiment.capitalize()} Sentiment')

# Creating subplots
fig, axes = plt.subplots(3, 1, figsize=(15, 10))

# Generating word clouds for each sentiment category
sentiments = df['sentiment'].unique()
for i, sentiment in enumerate(sentiments):
    generate_wordcloud(sentiment, axes[i])

plt.tight_layout()
plt.show()
```

• Word clouds are visual representations of text data where words are displayed in varying sizes, with the size of each word indicating its frequency or importance in the given text. While word clouds are a popular visualization technique, they are not typically used as a direct tool in natural language processing (NLP) for advanced analysis. Instead, they serve as a supplementary tool for visual exploration of textual data.

Here's how word clouds are commonly used in the context of NLP: o Data Exploration:

▪ Word clouds are often used in the initial stages of data exploration to visually inspect the most frequently occurring words in a corpus.

▪ This can help researchers or analysts get a quick overview of the dominant terms in the text data.

o Keyword Identification:

▪ Word clouds can be useful for identifying keywords or terms that stand out in a given document or set of documents.

▪ Analysts can quickly identify which words are more prominent in the dataset.

o Topic Visualization:

• In the context of topic modeling, word clouds can be generated for each identified topic to visualize the most relevant terms within that topic.

• This aids in interpreting and labeling topics based on the prevalent words.

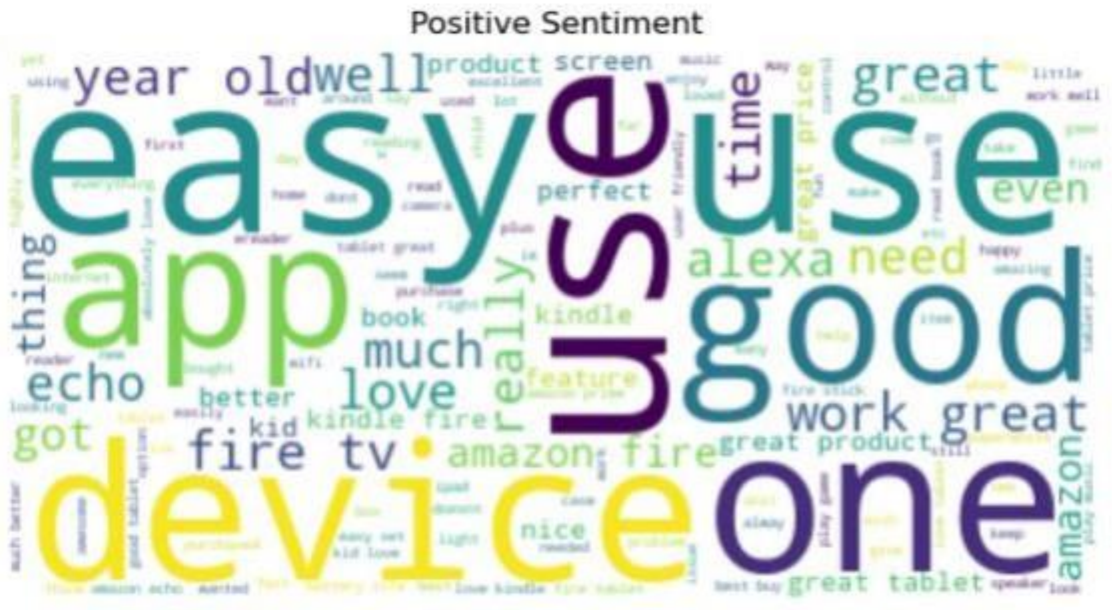
o Sentiment Analysis:

• Word clouds can be created for different sentiment categories (positive, negative, neutral) to visually represent the most common terms associated with each sentiment.

• This helps in understanding the language patterns in reviews or feedback.

• Word clouds are a starting point for visual exploration and interpretation, but they should be complemented with more advanced NLP methods for comprehensive analysis.

• The code defines a function “generate_wordcloud” that takes a sentiment category, combines the text data for that sentiment, and generates a word cloud. The subplots are then used to display word clouds for positive, negative, and neutral sentiments.



• Words that appear larger in the word cloud are those that occur more frequently in the positive sentiment context. These words are likely to be key indicators of positive sentiment in the dataset.

• Identify the most prominent and visually striking words in the word cloud. These are likely to be the top positive keywords that are commonly used in positive reviews or sentiments.

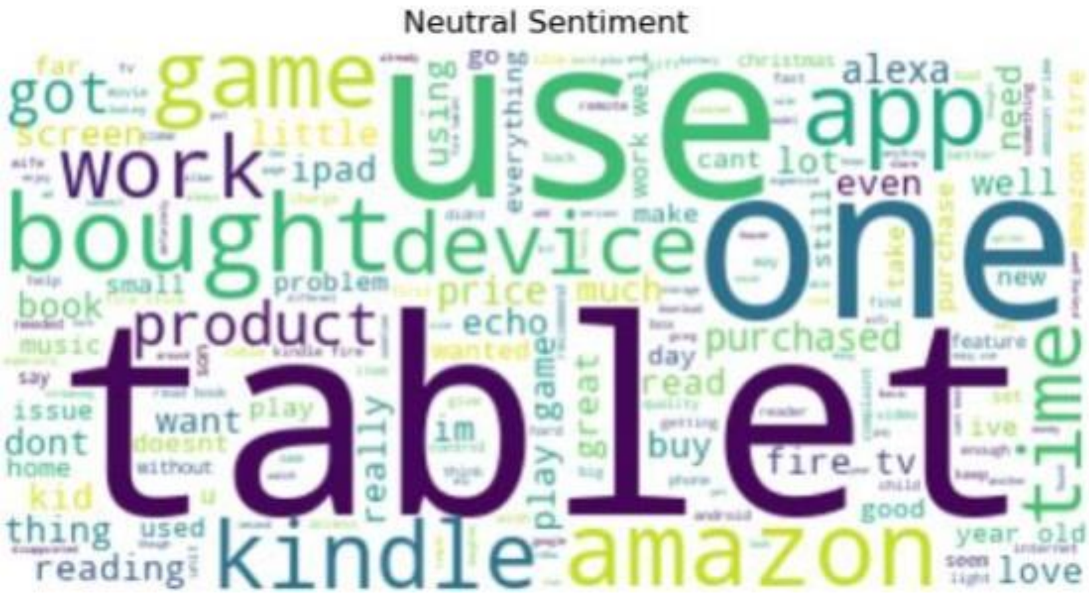
• Positive sentiment often correlates with positive feedback about specific product features or attributes. Look for words that describe the features, benefits, or characteristics of the product or service that customers appreciate.

• Positive sentiment word clouds may contain brand-related terms or affirmations. Customers expressing positive sentiments might mention the brand name, and associated products, or use positive adjectives that contribute to a favorable brand image.

• Considering the context in which positive words are used. Are they related to specific product categories, services, or experiences? Understanding the context provides deeper insights into the sources of positive sentiment.

• Beyond individual words, consider the emotional tone conveyed by the word cloud. Are there colors, shapes, or overall patterns that evoke a positive and uplifting feel? The visual aesthetics can contribute to the overall impact of the sentiment.

• Positive word clouds can be dynamic and change over time. Regularly updating and monitoring the word cloud helps businesses stay attuned to evolving customer sentiments and preferences.



• In a neutral sentiment word cloud, words should represent a balance between positive and negative sentiments. Look for words that do not carry strong emotional connotations and are commonly used in a more neutral context.

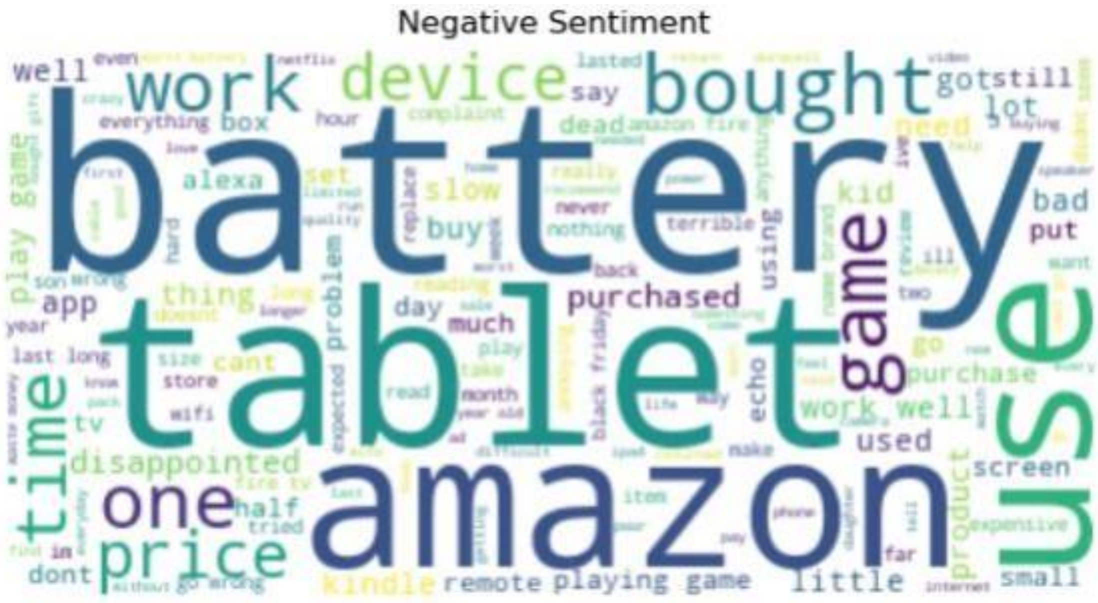
• Phrases or expressions frequently used in neutral sentiments might include terms like “average”, “standard”, “typical”, or other words that suggest a middle-ground opinion.

• Neutral sentiment may often be associated with descriptions of specific product features or attributes without expressing strong positive or negative opinions. Look for terms that describe in a factual and unbiased manner.

• Neutral sentiment word clouds may lack the heightened emotional tone observed in positive or negative sentiment word clouds.

• Words like “somewhat”, “partly”, or “occasionally” may appear frequently.

• Neutrality may involve discussing both positive and negative aspects without a clear preference.



• In a negative word cloud, terms like “poor”, “bad”, “unpleasant” or other strongly negative adjectives that signify dissatisfaction may appear.

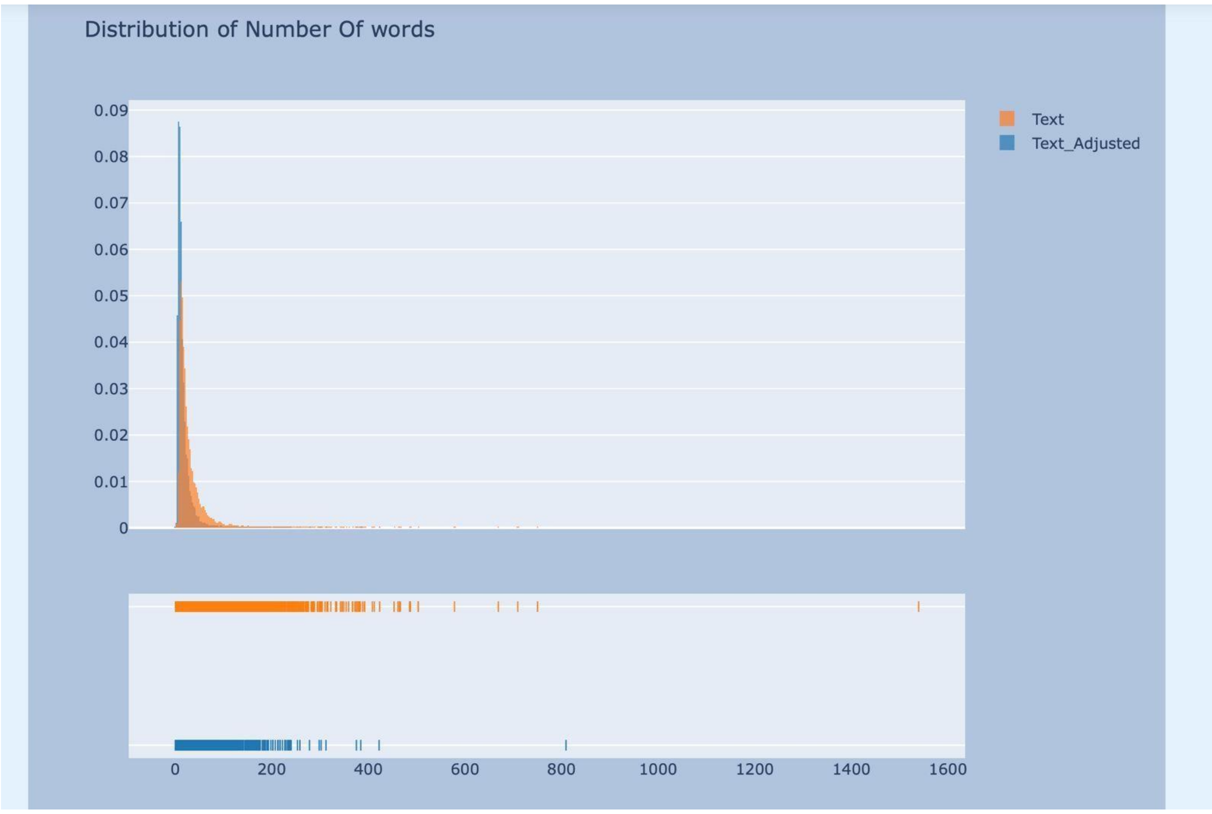
• Customers expressing dissatisfaction may use specific phrases to convey their disapointment or frustration.

• Terms related to defects, malfunstions, or shortcomings may be prevalent.

• Customers expressing negative sentiments might compare the product or service unfavorably to competitors. Look for mentions of competitor names or terms suggesting a preference for alternatives.

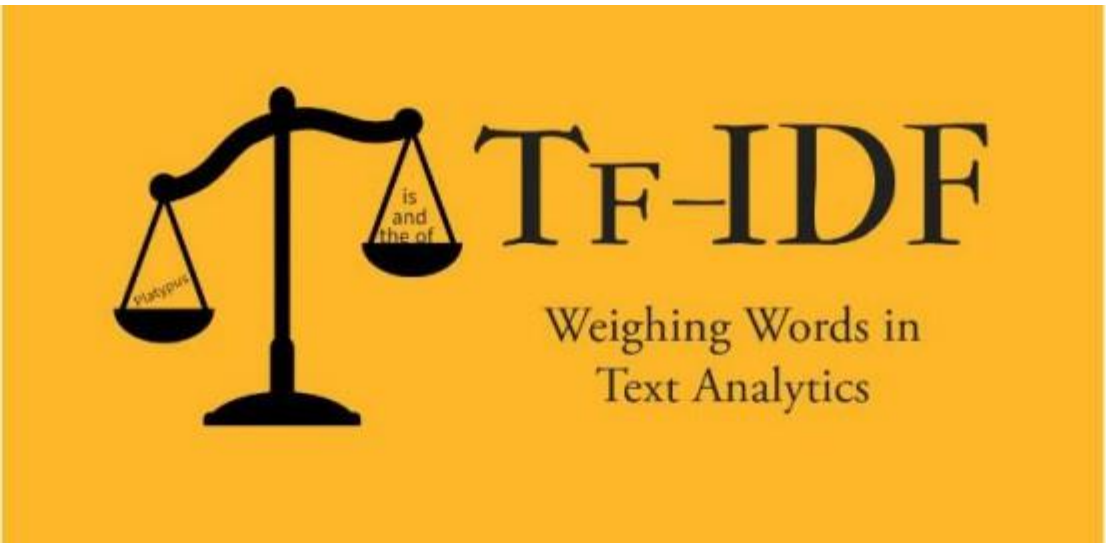
• Negative sentiment word clouds may highlight concerns about the quality of products or services. Customers may provide feedback on what needs to be fixed, enhanced, or addressed to enhance their experience.

• Negative sentiments in the word cloud may impact the brand image.




```
# logistic regression
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

text_data = df['reviews.text_adjusted']

# TF-IDF vectorization
tfidf_vectorizer = TfidfVectorizer()
X_tfidf = tfidf_vectorizer.fit_transform(text_data)

# train test split

X_train, X_test, y_train, y_test = train_test_split(X_tfidf, df['sentiment'], test_size=0.3, random_state=7)

# fitting the model

model = LogisticRegression()
model.fit(X_train, y_train)
predictions = model.predict(X_test)

# model evaluation

accuracy = accuracy_score(y_test, predictions)
classification_report_result = classification_report(y_test, predictions)

print("Accuracy:", accuracy)
print("Classification Report:", classification_report_result)
```



• The above code performs sentiment analysis using a logistic regression model with TF_IDF(Term Frequency-Inverse Document Frequency) vectorization.

• TF-IDF is often used in sentiment analysis as a feature representation technique to convert raw text into a numerical format that machine learning models can understand.

• Term Frequency(TF): Meausres how often a term appears in a specific document.

• Inverse Document Frequency(IDF): Measures the importance of a term across the entire corpus. Terms that are common across many documents receive lower IDF values.

• TF-IDF Calculation: TF-IDF is the product of TF and IDF. It gives a high weight to terms that are frequent in a specific document but rare across the entire corpus.

• TF-IDF feature matrix is used to train a sentiment analysis model. Commonly used models include logistic regression, support vector machines or deep learning models.

• During training, the model learns to associate certain TF-IDF patterns with positive, negative, or neutral sentiments.

• After training, the model can predict the sentiment of new, unseen text by transforming it using the same TF-IDF vectorizer and applying the trained model.

• Here, the text data is extracted from the “reviews.text_adjusted” column.TFIDF vectorization is applied using “TfidfVectorizer” from scikit-learn. This converts the text data into a matrix of TF-IDF features.

• The dataset is split into training and testing sets using “train test split” function.TF-IDF features(“X_tfidf) are the input, and the target variable is the “sentiment” column.

$$(Word{-}frequency{-}in{-}given{-}document) \cdot \log \frac{(Total - number - of - documents)}{(Number - of - documents - containing - word)}$$

Models

```
Accuracy: 0.9008563273073263
Classification Report:                precision    recall  f1-score   support

    negative       0.90       0.47       0.62       535
     neutral       0.64       0.39       0.48      1191
    positive       0.92       1.00       0.96      8784

    accuracy                             0.90     10510
   macro avg       0.82       0.62       0.69     10510
weighted avg       0.89       0.90       0.89     10510
```

• The above evaluation metrics (accuracy, precision, and F1-score) offer a comprehensive view of the performance of a sentiment analysis model.

• Accuracy Score: (90%) o High accuracy suggests that the model is overall effective in correctly classifying sentiments. o However, accuracy alone may not provide a complete picture, especially in imbalanced datasets.

• Precision:

o Negative Precision: 90%:

o Of all the instances predicted as negative, 90% are correctly classified as negative.

o High negative precision indicates the model's ability to avoid false negatives for negative sentiment.

o Neutral Precision: 64%:

o Of all instances predicted as neutral, 64% are correctly classified as neutral.

o Moderate neutral precision suggests some challenges in accurately identifying neutral sentiments.

o Positive Precision: 92%:

o Of all instances predicted as positive, 92% are correctly classified as positive.

o High positive precision indicates the model's ability to avoid false positives for positive sentiment.

• F1-Score:

o Negative F1-Score: 62%:

o The F1-score considers both precision and recall, providing a balance between false positives and false negatives.

o A lower negative F1-score indicates that the model may have challenges in achieving a balance between precision and recall for negative sentiment.

o Neutral F1-Score: 48% o The relatively low F1-score for neutral sentiment suggests challenges in achieving both high precision and recall for neutral predictions.

o Positive F1-Score: 96%

o A high positive F1-score indicates a good balance between precision and recall for positive sentiment.

• Inferences:

o Positive Sentiment:

o The model performs exceptionally well in identifying positive sentiment, with high precision (92%) and F1-score (96%).

o The high positive precision indicates that when the model predicts a positive sentiment, it is often correct.

o The high positive F1-score reflects a good balance between precision and recall for positive sentiment.

o Negative Sentiment:

o The model shows strong performance in negative sentiment with high precision (90%).

o However, the lower negative F1-score (62%) suggests room for improvement, indicating potential challenges in achieving a balance between precision and recall for negative sentiment.

o Neutral Sentiment:

o Identifying neutral sentiment appears to be more challenging for the model, as reflected in the lower precision (64%) and F1-score (48%).

o The model may struggle to achieve both high precision and recall for neutral predictions.

• Overall Model Assessment:

o While accuracy is high (90%), it's crucial to consider individual class metrics, especially in imbalanced datasets.

o Addressing challenges in neutral sentiment prediction and achieving a better balance between precision and recall for all classes could further improve the model.

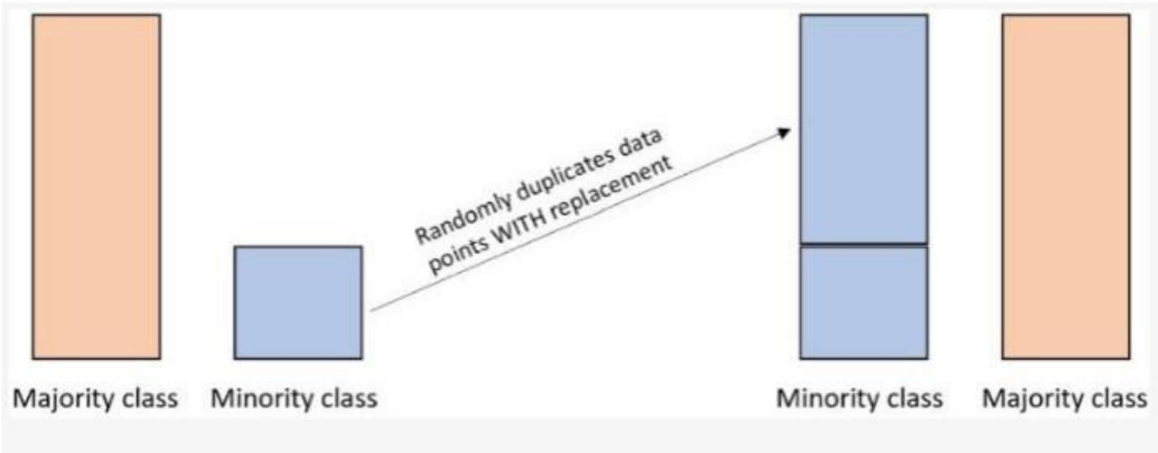
<u>Building a model using Logistic Regression using Random Over Sampling(ROS)</u>

```
# LOGISTIC USING ROS(RANDOM OVER SAMPLING)
```

```
from imblearn.over_sampling import RandomOverSampler
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

X = df['reviews.text_adjusted']
y = df['sentiment']

# TF-IDF vectorization
tfidf_vectorizer = TfidfVectorizer()
X_tfidf = tfidf_vectorizer.fit_transform(X)

# train test split
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.3, random_state=7)

# Oversampling the imblanced class using RandomOverSampler
oversampler = RandomOverSampler(sampling_strategy='auto', random_state=7)
X_train_resampled, y_train_resampled = oversampler.fit_resample(X_train, y_train)

# fitting the model with resampled x_train and y_train
model = LogisticRegression()
model.fit(X_train_resampled, y_train_resampled)
predictions = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, predictions)
classification_report_result = classification_report(y_test, predictions)

print("Accuracy:", accuracy)
print("Classification Report:", classification_report_result)
```

• The code snippet demonstrates a common practice for handling imbalanced datasets using oversampling and applying a logistic regression model .

Random Over Sampling:



Random Over Sampling:

• Random Oversampling is a technique used in the context of imbalanced datasets to address the issue where one class (usually the minority class) is underrepresented compared to the other class(es). In the case of sentiment analysis, this imbalance might occur when one sentiment class (e.g., negative sentiments) is less frequent than others.

• The Random Oversampling technique involves increasing the number of instances in the minority class by randomly duplicating existing instances until a more balanced distribution is achieved. The goal is to provide the machine learning model with a more equitable representation of each class during training.

• Here are the key steps involved in Random Oversampling:

• Identification of Imbalance:

o Recognize that there is an imbalance in the class distribution, and the minority class needs to be oversampled.

• Random Duplication:

o Randomly select instances from the minority class and duplicate them. This process is repeated until the desired balance between classes is achieved.

• Creation of a Balanced Dataset:

o Combine the original instances of both the minority and majority classes with the newly duplicated instances. This results in a new dataset with a more balanced class distribution.

• Training the Model:

o Train the machine learning model using this balanced dataset. The goal is to enable the model to better capture the patterns in the minority class.

• While Random Oversampling can be effective in mitigating class imbalance, it also has some potential drawbacks. Duplicating instances might lead to overfitting, as the model could memorize the duplicated samples. Additionally, does not introduce new information to the model, which might limit its ability to generalize to unseen data.

• It's essential to carefully evaluate the performance of the model on a separate test set to ensure that oversampling has improved the model's ability to correctly classify instances from the minority class without negatively impacting its performance on the majority class.

```
Accuracy: 0.90209324452902
Classification Report:              precision    recall  f1-score   support

     negative        0.70       0.78       0.74       535
      neutral        0.56       0.80       0.66      1191
     positive        0.99       0.92       0.95      8784

     accuracy                              0.90     10510
    macro avg        0.75       0.84       0.78     10510
 weighted avg        0.92       0.90       0.91     10510
```

• Accuracy: (90%)

o The model correctly predicted the sentiment of reviews 90% of the time.

o While accuracy is a commonly used metric, it may not provide a complete picture, especially in the presence of imbalanced classes. In sentiment analysis, where one sentiment class might dominate, a high accuracy score could be influenced by the majority class.

• Precision:

o Negative Precision: 70%

o Neutral Precision: 56%

o Positive Precision: 99%

o Precision measures the accuracy of the positive predictions made by the model for each sentiment class.

o A high precision for the positive class (99%) indicates that when the model predicts a review as positive, it is correct 99% of the time.

o Moderate precision for the negative and neutral classes suggests that the model's positive predictions are more reliable than its negative and neutral predictions.

• F1-Score:

o Negative F1-Score: 74%

o Neutral F1-Score: 66%

o Positive F1-Score: 95%

o F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics.

o The F1-scores for positive and negative classes are relatively high, indicating a good balance between precision and recall for these classes.

o The lower F1-score for the neutral class suggests a trade-off between precision and recall, indicating room for improvement in predicting neutral sentiments.

• Overall:

o The model performs exceptionally well in identifying positive sentiments, as indicated by high precision and F1-score.

o The model's ability to correctly predict negative and neutral sentiments is comparatively weaker, as reflected in lower precision and F1-score for these classes.

o The imbalance in precision scores suggests a potential bias in the model towards the majority class (positive sentiments), highlighting the need for further fine-tuning, feature engineering, or adjusting class weights.

o In summary, while the high accuracy is promising, a more nuanced analysis of precision and F1-score reveals areas for improvement, particularly in achieving a better balance across all sentiment classes.

Building a model using Gradient Boosting Classifier using Random Over Sampling(ROS)

```
# GRADIENT BOOSTING CLASSIFIER USING ROS(RANDOM OVER SAMPLING)
```

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, classification_report
from imblearn.over_sampling import RandomOverSampler
from imblearn.pipeline import make_pipeline

# train_test_Split
X_train, X_test, y_train, y_test = train_test_split(df['reviews.text_adjusted'], df['sentiment'], test_size=0.3, random_state=7)

# TF-IDF vectorization
tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

# pipeline of RandomOverSampler  and GradientBoostingClassifier
model = make_pipeline(RandomOverSampler(), GradientBoostingClassifier(random_state=7))

# fit the model
model.fit(X_train_tfidf, y_train)
predictions = model.predict(X_test_tfidf)

# Evaluate the model
accuracy = accuracy_score(y_test, predictions)
classification_report_result = classification_report(y_test, predictions)

print("Accuracy:", accuracy)
print("Classification Report:")
print(classification_report_result)
```

• The above code implements a machine learning pipeline for sentiment analysis using a combination of Random OverSampler and a GradientBoostingClassifier.

• A pipeline in mechine elarning is a way to streamline a lot of the routine processes, making it easier to keep the code organized, reduce errors, and simplify the model deployment process.

• Gradient Boosting is an ensemble learning technique that builds a series of weak learners(usually decision trees) sequentially, with each tree correcting the errors of it predecessor.

• The final prediction is made by summing the predictors of all the waek learners. Each tree contributes to the overall prediction , and the combination of these weak learners creates a strong predictive model.

```
Accuracy: 0.8182683158896289
Classification Report:
              precision    recall  f1-score   support

    negative       0.62      0.69      0.65       535
     neutral       0.37      0.74      0.49      1191
    positive       0.98      0.84      0.90      8784

    accuracy                           0.82     10510
   macro avg       0.65      0.76      0.68     10510
weighted avg       0.89      0.82      0.84     10510
```

• Accuracy: 0.81 (81%)

o The model correctly predicted the sentiment of reviews 81% of the time.

o While accuracy is a commonly used metric, it may not provide a complete picture, especially in the presence of imbalanced classes. In sentiment analysis, where certain sentiments may dominate, a high accuracy score could be influenced by the majority class.

• Precision:

o Negative Precision: 62%

o Neutral Precision: 37%

o Positive Precision: 98%

o Precision measures the accuracy of the positive predictions made by the model for each sentiment class.

o A high precision for the positive class (98%) indicates that when the model predicts a review as positive, it is correct 98% of the time.

o The lower precision for negative and neutral classes suggests that the model's positive predictions are more reliable than its negative and neutral predictions.

• F1-Score: 41

o Negative F1-Score: 69%

o Neutral F1-Score: 74%

o Positive F1-Score: 84%

o F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics.

o The F1-scores for neutral and positive classes are relatively high, indicating a good balance between precision and recall for these classes.

o The lower F1-score for the negative class suggests a trade-off between precision and recall, indicating room for improvement in predicting negative sentiments.

• Overall :

o The model performs exceptionally well in identifying positive sentiments, as indicated by high precision and F1-score.

o The model's ability to correctly predict negative and neutral sentiments is comparatively weaker, as reflected in lower precision and F1-score for these classes.

o The imbalance in precision scores suggests a potential bias in the model towards the majority class (positive sentiments), highlighting the need for further fine-tuning, feature engineering, or adjusting class weights.

Building a model using Random Forest using Random Over Sampling(ROS)

```
# RANDOM FOREST USING ROS(RANDOM OVER SAMPLING)
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import train_test_split
from imblearn.over_sampling import RandomOverSampler
from sklearn.feature_extraction.text import TfidfVectorizer

# train_test_Split
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, df['sentiment'], test_size=0.3, random_state=7)

# Applying random oversampling
ros = RandomOverSampler(random_state=42)
X_resampled, y_resampled = ros.fit_resample(X_train, y_train)

# Random Forest Classifier
rf_model = RandomForestClassifier(random_state=42)

# fit the model
rf_model.fit(X_resampled, y_resampled)
predictions = rf_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, predictions)
classification_report_result = classification_report(y_test, predictions)

print("Random Forest Classifier Results:")
print("Accuracy:", accuracy)
print("Classification Report:\n", classification_report_result)
```

• Random Forest is an ensemble learning technique that builds multiple decision trees during training and merges them to get more accurate and stable predictions. It belongs to bagging algorithms.

• Random Forest employs bootstrap sampling, meaning that each tree is trained on a random sample of the training data with replacement. This creates multiple diverse subsets of the data.

• By combining the predictions of multiple trees, Random Forest tends to reduce overfitting and improve generalization performance. It is less sensitive to noise and outliers compared to individual decision trees.

```
Random Forest Classifier Results:
Accuracy: 0.8961941008563273
Classification Report:
              precision    recall  f1-score   support

    negative       0.93      0.64      0.76       535
     neutral       0.68      0.33      0.45      1191
    positive       0.91      0.99      0.95      8784

    accuracy                           0.90     10510
   macro avg       0.84      0.65      0.72     10510
weighted avg       0.88      0.90      0.88     10510
```

• Accuracy: (89%)

o The model correctly predicted the sentiment of reviews 89% of the time.

o A high accuracy score suggests that the model is generally effective in making correct predictions across all sentiment classes.

• Precision:

o Negative Precision: 93%

o Neutral Precision: 68%

o Positive Precision: 91%

o Precision measures the accuracy of the positive predictions made by the model for each sentiment class.

o High precision for negative and positive classes indicates that when the model predicts a review as negative or positive, it is correct 93% and 91% of the time, respectively.

o The lower precision for the neutral class (68%) suggests that the model's neutral predictions are less reliable than its negative and positive predictions.

• F1-Score:

o Negative F1-Score: 64%

o Neutral F1-Score: 33%

o Positive F1-Score: 99%

o F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics.

o The high F1-score for the positive class indicates a good balance between precision and recall for positive sentiments.

o The lower F1-scores for negative and neutral classes suggest challenges in achieving a balance between precision and recall for these classes.

• Overall :

o The model performs exceptionally well in identifying positive sentiments, as indicated by high precision and F1-score.

o The model's ability to correctly predict negative sentiments is strong, reflected in high precision.

o The model's performance on neutral sentiments is comparatively weaker, as indicated by lower precision and F1-score for the neutral class.

o The extremely high F1-score for the positive class suggests that the model is robust in capturing positive sentiments with high precision and recall.

Building a model using Naï ve Bayes using Random Over Sampling(ROS)

```
# NAIVE BAYES USING ROS(RANDOM OVER SAMPLING)
```

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report
from imblearn.over_sampling import RandomOverSampler
from imblearn.pipeline import make_pipeline

#  train_test_Split
X_train, X_test, y_train, y_test = train_test_split(df['reviews.text_adjusted'], df['sentiment'], test_size=0.3, random_state=7)

# TF-IDF vectorization
tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

# pipeline of RandomOverSampler  and Naive Bayes
model = make_pipeline(RandomOverSampler(), MultinomialNB())

# fit the model
model.fit(X_train_tfidf, y_train)
predictions = model.predict(X_test_tfidf)

# Evaluate the model
accuracy = accuracy_score(y_test, predictions)
classification_report_result = classification_report(y_test, predictions)

print("Accuracy:", accuracy)
print("Classification Report:")
print(classification_report_result)
```

• The above code implements sentiment analysis using Naï ve Bayes, specifically Multinomial Naï ve Bayes algorithm.

• Naï ve Bayes is a family of probabilistic classification alogorithms based on Baye’s theorem. Despite its “naï ve” assumption of independence between features , the Naï ve Bayes classifier has proven to be suprisingly effective in various real_world applications, particularly in NLP tasks like text classification.

• In context of text classification, Naï ve Bayes is commonly used to predict the category of a document or the sentiment of a piece of text.

• Top applications of Naï ve Bayes are spam filtering, sentiment analysis, topoc classification in text classification.

• The provided evaluation metrics (accuracy, precision, and F1-score) offer insights into the performance of a sentiment analysis model. Let's interpret each metric in detail:

```
Accuracy: 0.8114176974310181
Classification Report:
              precision    recall  f1-score   support

    negative       0.52      0.67      0.59       535
     neutral       0.34      0.56      0.43      1191
    positive       0.95      0.85      0.90      8784

    accuracy                           0.81     10510
   macro avg       0.61      0.70      0.64     10510
weighted avg       0.86      0.81      0.83     10510
```

• Accuracy: (81%)

o The model correctly predicted the sentiment of reviews 81% of the time.

o A high accuracy score suggests that the model is generally effective in making correct predictions across all sentiment classes.

• Precision:

o Negative Precision: 52%

o Neutral Precision: 34%

o Positive Precision: 95%

o Precision measures the accuracy of the positive predictions made by the model for each sentiment class.

o High precision for the positive class (95%) indicates that when the model predicts a review as positive, it is correct 95% of the time.

o The lower precision for negative and neutral classes (52% and 34%, respectively) suggests that the model's predictions for these classes may have a higher rate of false positives.

• F1-Score:

o Negative F1-Score: 67%

o Neutral F1-Score: 56%

o Positive F1-Score: 85%

o F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics.

o The F1-scores reflect a balance between precision and recall for each sentiment class.

o The lower F1-scores for negative and neutral classes suggest challenges in achieving a balance between precision and recall for these classes.

• Overall :

o The model performs well in identifying positive sentiments, as indicated by high precision and F1-score.

o The model's ability to correctly predict negative sentiments is moderate, reflected in a lower precision but a reasonably high F1-score. o The model's performance on neutral sentiments is comparatively weaker, as indicated by lower precision and F1-score for the neutral class.

Building a model using Support Vector Classifier using Random Over Sampling(ROS)

```
# SUPPORT VECTOR CLASSIFIER USING ROS(RANDOM OVER SAMPLING)
```

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
from imblearn.over_sampling import RandomOverSampler
from imblearn.pipeline import make_pipeline

# train_test_Split
X_train, X_test, y_train, y_test = train_test_split(df['reviews.text_adjusted'], df['sentiment'], test_size=0.3, random_state=7)

# TF-IDF vectorization
tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

#  pipeline of RandomOverSampler and SVM
model = make_pipeline(RandomOverSampler(), SVC())

# fit the model
model.fit(X_train_tfidf, y_train)
predictions = model.predict(X_test_tfidf)

# Evaluate the model
accuracy = accuracy_score(y_test, predictions)
classification_report_result = classification_report(y_test, predictions)

print("Accuracy:", accuracy)
print("Classification Report:")
print(classification_report_result)
```

• The above code implements a sentiment analysis model using a Support Vector Machine(SVM) classifier with TF_IDF vectorization and random over sampling.

• A Support Vector Classifier(SVC), also known as Support Vector Machine(SVM) for classification is a supervised machine learning algorithm. SVM's are effective in high-dimensional spaces and are especially well-suited for tasks where clear boundaries exist between different classes.

• The primary goal of an SVC is to find the optimal hyperplane that seperates different classes in a feature space.

• The optimal hyperplane is the one that maximizes the margin, which is the distance between the hyperplane and the nearest data point from either class.

• Support vectors are the data points that lie closest to the decision boundary(the hyperplane). These points are crucial for defining the optimal hyperplane and hence, the classifier.

• The use of a margin and regularization parameter helps prevent overfitting.

• SVM are widely used for sentiment analysis and topic categorization for text classification.

```
Accuracy: 0.9310180780209324
Classification Report:
              precision    recall  f1-score   support

    negative       0.93      0.65      0.77       535
     neutral       0.73      0.64      0.68      1191
    positive       0.95      0.99      0.97      8784

    accuracy                           0.93     10510
   macro avg       0.87      0.76      0.81     10510
weighted avg       0.93      0.93      0.93     10510
```

• Accuracy: (93%)

o The model correctly predicted the sentiment of reviews 93% of the time.

o A high accuracy score suggests that the model is generally effective in making correct predictions across all sentiment classes.

• Precision:

o Negative Precision: 93%

o Neutral Precision: 73%

o Positive Precision: 95%

o Precision measures the accuracy of the positive predictions made by the model for each sentiment class.

o High precision for the positive and negative classes (95% and 93%, respectively) indicates that when the model predicts a review as positive or negative, it is correct 95% and 93% of the time.

o The precision for the neutral class is also good at 73%.

• F1-Score:

o Negative F1-Score: 77%

o Neutral F1-Score: 68%

o Positive F1-Score: 97%

o F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics.

o The F1-scores reflect a balance between precision and recall for each sentiment class.

o The high F1-scores for positive and negative classes (97% and 77%, respectively) indicate a good balance between precision and recall.

o The F1-score for the neutral class is also reasonable at 68%.

• Overall

o The model exhibits high accuracy, indicating strong overall predictive performance.

o Precision scores are high for all sentiment classes, suggesting that the model's positive, negative, and neutral predictions are accurate.

o F1-scores reflect a good balance between precision and recall, especially for positive and negative sentiments.

Model	Accuracy
Logistic Regression (Using ROS)	90%
Gradient Boosting Classifier	82 %
Random Forest Classifier	90 %
Naïve Bayes (Multinomial NB)	81%
Support Vector Machine	93%

• Comparing the accuracy results of different models provides insights into their performance on the sentiment analysis task.

• Logistic Regression (Using ROS) - 90% Accuracy:**

• The logistic regression model, using Random OverSampling (ROS) to address class imbalance, achieves a commendable 90% accuracy. This model appears to handle the sentiment analysis task well.

• Gradient Boosting Classifier - 82% Accuracy:

• The gradient boosting classifier demonstrates good performance with an 82% accuracy. While slightly lower than the logistic regression model, gradient boosting might excel in capturing complex relationships within the data.

• Random Forest Classifier - 90% Accuracy:

• The random forest classifier achieves the same accuracy as logistic regression (90%). Random forests are known for their robustness and ability to handle various types of data, making it a strong contender for sentiment analysis.

• Naï ve Bayes (Multinomial NB) - 81% Accuracy:

• The Naï ve Bayes model, specifically Multinomial Naï ve Bayes, achieves an accuracy of 81%. While not as high as some other models, Naï ve Bayes is known for its simplicity and efficiency in text classification tasks.

• Support Vector Machine - 93% Accuracy:

• The Support Vector Machine (SVM) stands out with the highest reported accuracy of 93%. SVMs are powerful for tasks with complex decision boundaries, and in this case, it performs exceptionally well in sentiment analysis.

• Overall Comparison and Considerations:

▪ The SVM model demonstrates the highest accuracy, suggesting it is well-suited for the sentiment analysis task in this context.

▪ Logistic regression and the random forest classifier also perform well, sharing the top position with 90% accuracy.

▪ Gradient boosting, while slightly lower in accuracy, may still provide valuable insights, especially if interpretability is crucial.

▪ Naï ve Bayes, with its simplicity, achieves a respectable accuracy of 81%, making it a viable option for certain applications.

Model	Sentiment	F1-score	Precision
Logistic Regression (Using ROS)	Negative	0.70	0.74
	Neutral	0.56	0.66
	Positive	0.99	0.95
Gradient Boosting Classifier	Negative	0.62	0.65
	Neutral	0.37	0.49
	Positive	0.98	0.90
Random Forest Classifier	Negative	0.93	0.76
	Neutral	0.68	0.45
	Positive	0.91	0.95
Naïve Bayes (Multinomial NB)	Negative	0.52	0.59
	Neutral	0.34	0.43
	Positive	0.95	0.90
Support Vector Machine	Negative	0.93	0.77
	Neutral	0.73	0.68
	Positive	0.95	0.97

• Logistic Regression (Using ROS):

• F1-score: Achieves high F1-scores across all sentiments, indicating balanced precision and recall.

• Precision: Particularly strong precision for positive sentiment (0.95).

• Gradient Boosting Classifier:

• F1-score: Lowest F1-score for neutral sentiment (0.37), indicating challenges in classifying neutral sentiment.

• Precision: Positive sentiment precision is relatively high (0.90).

• Random Forest Classifier:

• F1-score: Strong F1-scores across all sentiments, especially high for negative sentiment (0.93).

• Precision: Highest precision for negative sentiment (0.76).

• Naï ve Bayes (Multinomial NB):

• F1-score: Lowest F1-scores among all models, particularly for neutral sentiment (0.34).

• Precision: Positive sentiment precision is relatively high (0.90).

• Support Vector Machine:

• F1-score: High F1-scores across all sentiments, indicating balanced performance.

• Precision: Strong precision for all sentiments, particularly high for positive sentiment (0.97).

Overall Comparison and Considerations:

Logistic Regression (Using ROS):

o Balanced performance across sentiments with high precision for positive sentiment.

Gradient Boosting Classifier:

o Struggles with neutral sentiment classification (low F1-score).

o Strong positive sentiment precision.

Random Forest Classifier:

o Overall strong performance with high F1-scores and precision.

o Particularly high precision for negative sentiment.

Naï ve Bayes (Multinomial NB):

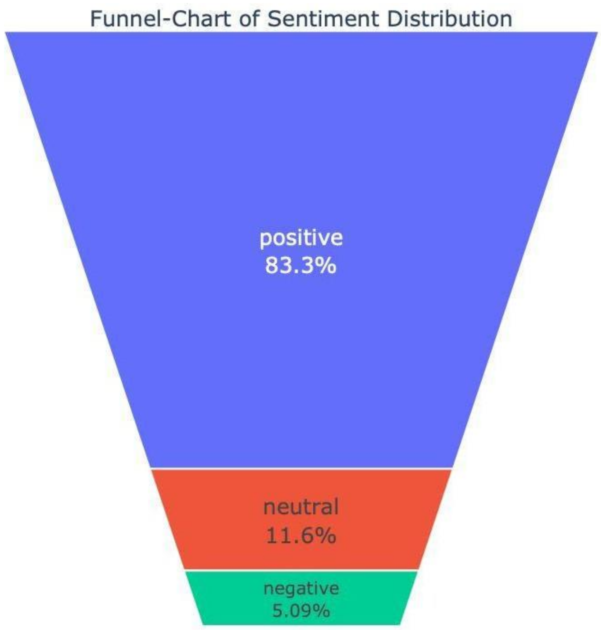
o Lower F1-scores compared to other models.

o High precision for positive sentiment.

o Support Vector Machine:

o Balanced and high-performance across sentiments.

EDA



•A funnel chart is a type of chart often used to visualize stages of a process . It resembles a funnel, wider at the top and narrower at the bottom, with each section representing a polarity. The width of each section corresponds to the quantity or percentage of items at that stage.The funnel chart visually represents the distribution of sentiment labels in the “sentiment” column.

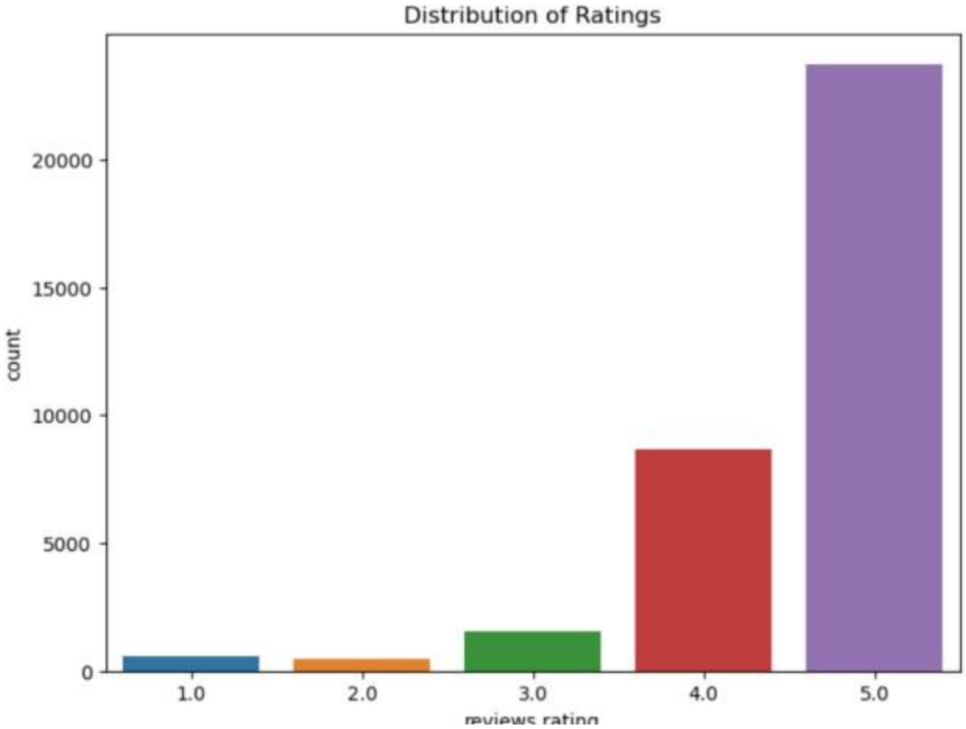
• Each section corresponds to a sentiment category(“positive”, “negative”, “neutral”).

• Approximately 83.3% of the sentiments in the dataset are categorized as positive. This indicates a predominant positive sentiment in the analyzed texts.

• Around 5.1% of the sentiments are categorized as negative, suggesting a relatively lower occurrence of negative sentiments.

• And Approximately 7% of the sentiments are categorized as neutral, indicating instances where the sentiment polarity is close to zero.

```
plt.figure(figsize=(8, 6))
sns.countplot(x='reviews.rating', data=df)
plt.title('Distribution of Ratings')
plt.show()
```



• The count plot visualizes the distribution of ratings in the “review.rating” column

• The X-axis represents the unique rating values, and the Y-axis represents the count of each rating.

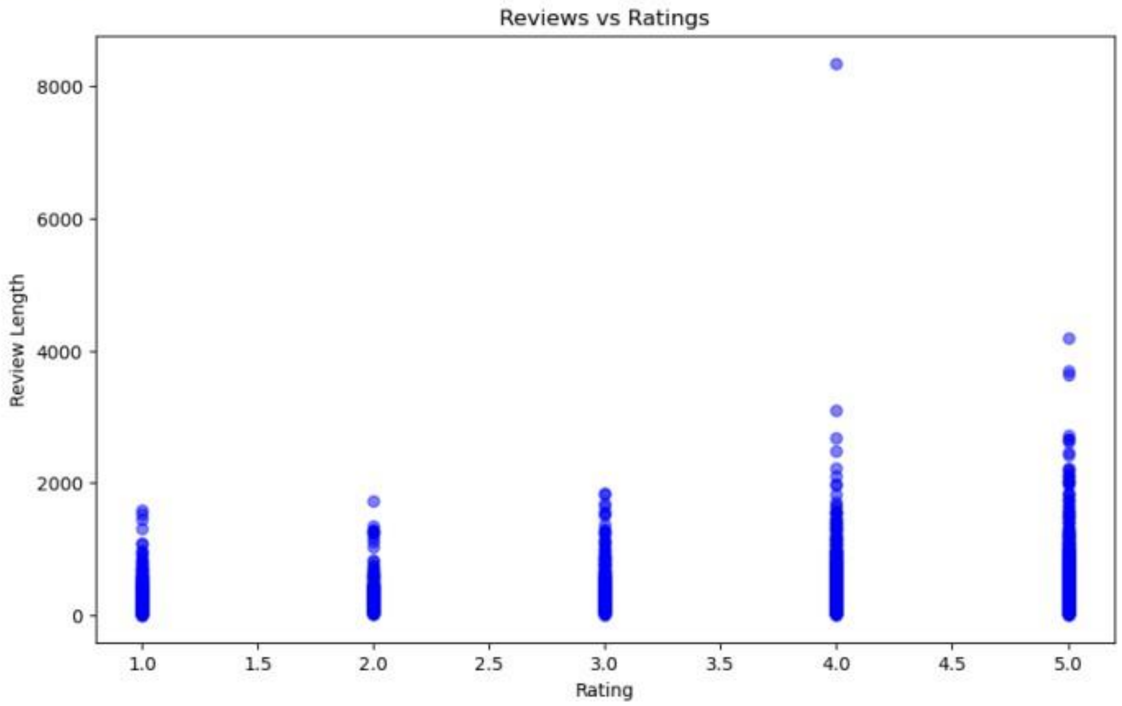
• A substantial number of reviews have received the highest rating of 5. Similarly, a considerbale number of reviews have received a rating of 4.

• The count plot doesn’t show the distribution of lower ratings explicitly, but it implies that the frequency of lower rating(e.g. 1,2 and 3) is comparitively lower than the higher ratings.

• The high concentration of ratings around 4 and 5 suggests a generally positive sentiment in the reviews.

• The visualization provides a quick overview of the distribution of ratings and can be useful for understanding the overall satisfaction level based on customer or user reviews.

```
df['review_length'] = df['reviews.text'].apply(len)
plt.figure(figsize=(10, 6))
plt.scatter(df['reviews.rating'], df['review_length'], alpha=0.5, color='blue')
plt.title('Reviews vs Ratings')
plt.xlabel('Rating')
plt.ylabel('Review Length')
plt.show()
```



• The scatter plot illustrates the relationship between the length of reviews and the ratings given by reviewers.

• The scattered points on the plot reveal the diversity of review lengths across different ratings.

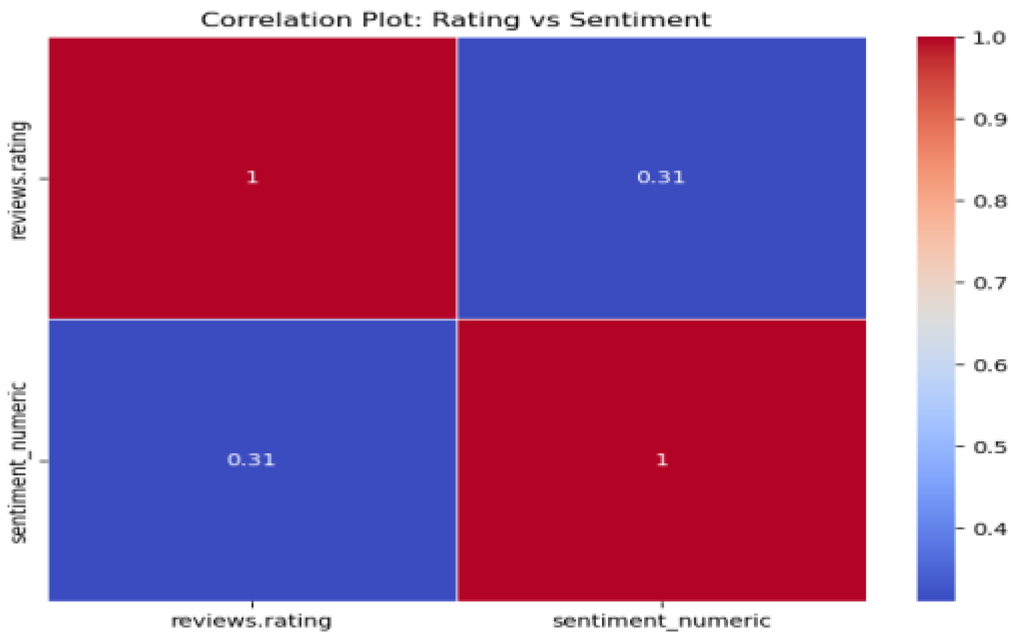
• The concentration of points in specific regions of the plot can indicate trends or patterns.

• There is a discernible pattern, it might be visible as clusters or trends in the scatter plot.

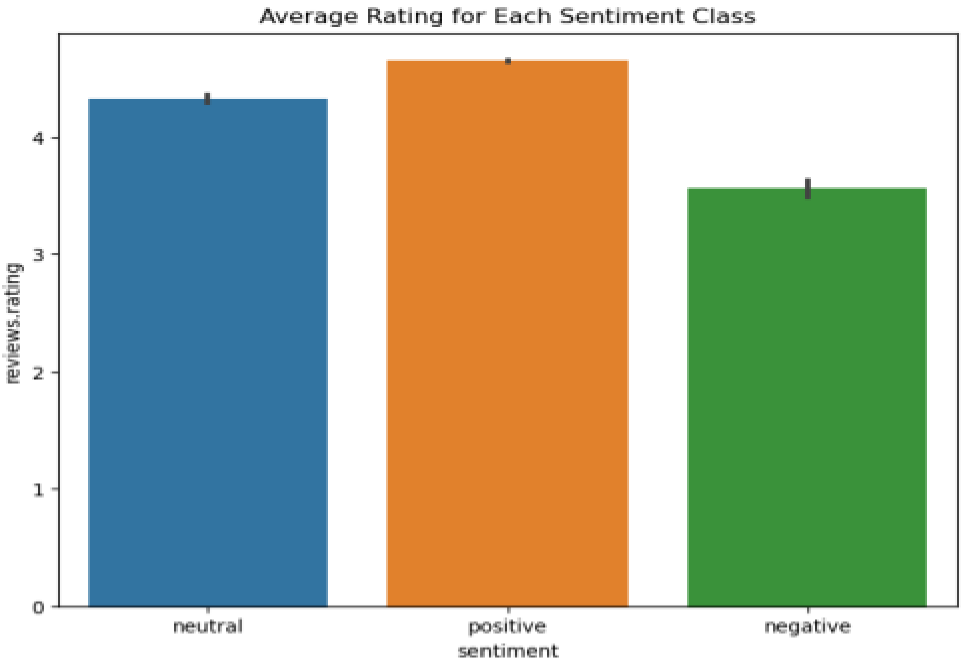
• Reviewers may exhibit different behaviors based on the length of their reviews and the ratings they assign.

• High-density areas or trends in the plot could provide insights into whether there is a correlation between the length of reviews and the ratings given.

• It helps analysts and stakeholders understand potential correlations or patterns in how the length of reviews may be associated with the ratings assigned by reviewers.



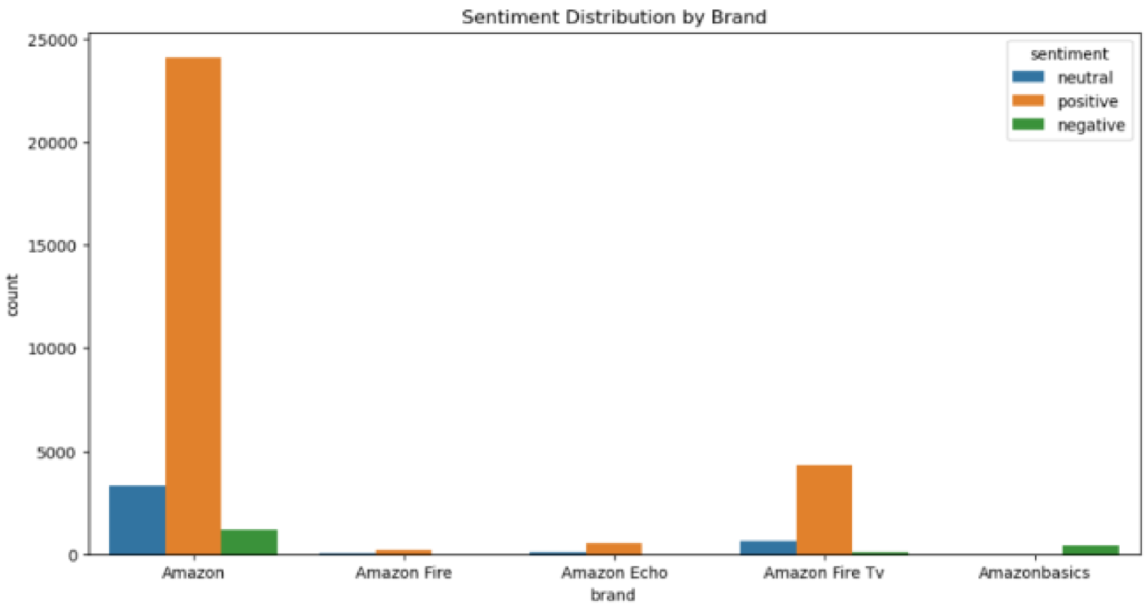
- A new column named `sentiment_numeric` is created by mapping the categorical sentiment values ('negative', 'neutral', 'positive') to numeric values (-1, 0, 1) using the `sentiment_mapping` dictionary.
- A correlation matrix is computed for the columns 'reviews.rating' and 'sentiment_numeric'.
- The correlation coefficient quantifies the strength and direction of a linear relationship between two variables, ranging from -1 to 1. A positive correlation suggests a positive relationship, while a negative correlation suggests an inverse relationship. - The correlation plot visually depicts the correlation between the numeric representation of sentiment and the ratings.
- Positive correlation values suggest that as one variable increases, the other tends to increase as well, while negative values suggest an inverse relationship.
- The heatmap allows for quick identification of patterns or trends in the correlation between sentiment and ratings.



• The bar plot provides a straightforward comparison of average ratings across different sentiment classes.

• It allows for a quick assessment of the relationship between sentiment and average ratings.

• The visualization helps to understand how sentiment classes are distributed in terms of average ratings, offering insights into how sentiment may correlate with the overall positivity or negativity of reviews in the dataset.



• The plot enables a brand-specific view of sentiment composition, showing how reviews are distributed among different sentiment categories for each brand.

• Brands with higher positive sentiment bars may indicate a generally positive customer sentiment, while higher negative sentiment bars suggest more critical reviews.

• The count of reviews for each brand provides insights into the popularity or customer engagement level.

• Combining this information with sentiment distribution allows for a more comprehensive understanding of brand perception.

• The visualization facilitates a quick assessment of sentiment distribution across brands, helping stakeholders identify patterns or trends associated with customer sentiment.

• It aids in identifying potential areas for improvement, understanding customer satisfaction, and making informed brand-related decisions based on customer feedback.

STEP BY STEP WALKTHROUGH

Data Cleaning

Data cleaning involves identifying and correcting errors or inconsistencies in the dataset.

- Tasks include handling missing values, correcting data formats, removing duplicates, and addressing outliers.
- The goal is to ensure that the data is accurate, consistent, and suitable for analysis.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35631 entries, 0 to 35630
Data columns (total 7 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Unnamed: 0           35631 non-null  int64  
 1   brand                35631 non-null  object 
 2   manufacturer         35631 non-null  object 
 3   reviews.doRecommend  35037 non-null  object 
 4   reviews.rating       35598 non-null  float64
 5   reviews.text         35630 non-null  object 
 6   reviews.title        35626 non-null  object 
dtypes: float64(1), int64(1), object(5)
memory usage: 1.9+ MB
```

	brand	manufacturer	reviews.doRecommend	reviews.rating	reviews.text	reviews.title	reviews.text_adjusted	sentiment	jaccard_score	Num_words_RTA
0	Amazon	Amazon	True	5.0	This product so far has not disappointed. My c...	Kindle	product far disappointed child love use like a...	neutral	0.344828	13
1	Amazon	Amazon	True	5.0	great for beginner or experienced person. Boug...	very fast	great beginner experienced person bought gift ...	positive	0.312500	7
2	Amazon	Amazon	True	5.0	Inexpensive tablet for him to use and learn on...	Beginner tablet for our 9 year old son.	inexpensive tablet use learn , step nabi thril...	positive	0.259259	12
3	Amazon	Amazon	True	4.0	I've had my Fire HD 8 two weeks now and I love...	Good!!!	ive fire hd 8 two week love tablet great value...	positive	0.447917	60
4	Amazon	Amazon	True	5.0	I bought this for my grand daughter when she c...	Fantastic Tablet for kids	bought grand daughter come visit set user , en...	positive	0.336735	53

```
df.isnull().sum()
```

```
Unnamed: 0               0
brand                    0
manufacturer             0
reviews.doRecommend    594
reviews.rating          33
reviews.text             1
reviews.title            5
dtype: int64
```

```
df.dropna(subset=['reviews.doRecommend','reviews.rating','reviews.text','reviews.title'],inplace=True)
```

```
df.isnull().sum()
```

```
Unnamed: 0             0
brand                  0
manufacturer           0
reviews.doRecommend    0
reviews.rating         0
reviews.text           0
reviews.title          0
dtype: int64
```

```
df.drop('Unnamed: 0',axis=1,inplace=True)
```

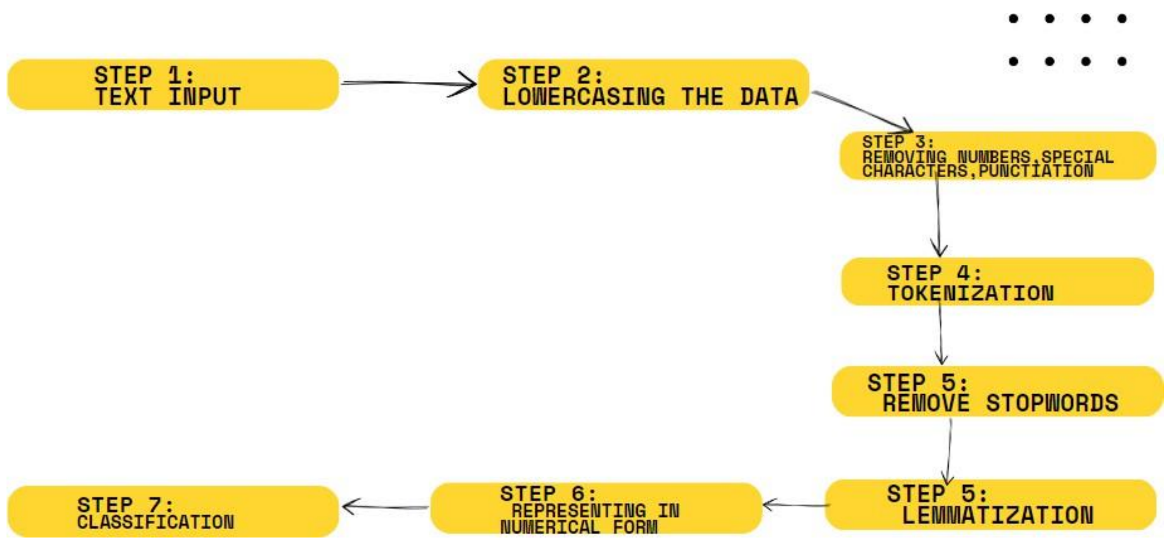
```
df
```

	brand	manufacturer	reviews.doRecommend	reviews.rating	reviews.text	reviews.title
0	Amazon	Amazon	True	5.0	This product so far has not disappointed. My c...	Kindle
1	Amazon	Amazon	True	5.0	great for beginner or experienced person. Boug...	very fast
2	Amazon	Amazon	True	5.0	Inexpensive tablet for him to use and learn on...	Beginner tablet for our 9 year old son.
3	Amazon	Amazon	True	4.0	I've had my Fire HD 8 two weeks now and I love...	Good!!!
4	Amazon	Amazon	True	5.0	I bought this for my grand daughter when she c...	Fantastic Tablet for kids
...	...	...	...	...	...	...
35626	Amazon	Amazon	True	3.0	Unable to download the Instagram app. which st...	The camera is awful!
35627	Amazon	Amazon	True	5.0	Pros: 5 Ghz wifi, supports expansion cardsCons...	Very good for reading, email, internet
35628	Amazon	Amazon	True	5.0	Bought it for my little grandson he likes game...	very good tablet
35629	Amazon	Amazon	True	5.0	I just like reading amd playing games on it. B...	Works great
35630	Amazon	Amazon	True	4.0	Allows for reading of my Kindle lib and even p...	Works as advertised

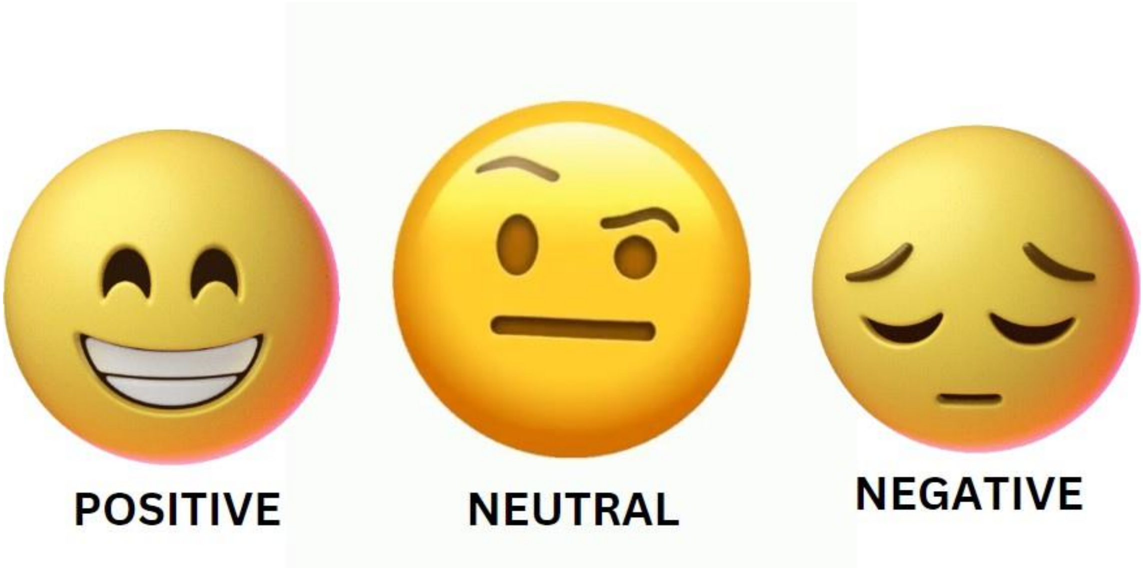
35033 rows × 6 columns

Data Preprocessing:

- Data pre-processing involves transforming raw data into a format suitable for machine learning algorithms.
- Tasks may include feature scaling, feature encoding (e.g., one-hot encoding for categorical variables), and feature engineering to create new features.
- The goal is to prepare the data in a way that enhances the performance of machine learning models.



Identifying the Traget Variable:



Based on the polarity scores we classify them as positive,negative and neutral, creating a new column which is the target variable.

- The target variable, also known as the dependent variable, is the variable we are trying to predict or understand.
- Identifying the target variable is crucial as it defines the objective of the machine learning problem.
- For example, in a predictive model for house prices, the target variable would be the sale price of houses.

Model Building:

Base models refer to the simplest or most basic versions of machine learning algorithms before any modifications or enhancements are applied. These models serve as the foundation upon which more complex algorithms are built or as benchmarks for comparison. Here are examples of base models for commonly used machine learning algorithms:

Linear Regression:

Base Model: Simple linear regression, where a single independent variable is used to predict a continuous dependent variable.

Description: Fits a linear relationship between the independent and dependent variables.

Logistic Regression:

Base Model: Binary logistic regression, where the dependent variable is binary (e.g., 0 or 1).

Description: Estimates the probability that an instance belongs to a particular class using a logistic function.

Decision Trees:

Base Model: A single decision tree without any pruning or regularization.

Description: Divides the feature space into regions, with each region corresponding to a leaf node in the tree, based on a series of binary decisions.

K-Nearest Neighbors (KNN):

Base Model: KNN with a default value of k (number of nearest neighbors) and using Euclidean distance as the similarity metric.

Description: Classifies instances based on the majority class of their nearest neighbors in the feature space.

Support Vector Machines (SVM):

Base Model: SVM with linear kernel and default hyperparameters.

Description: Constructs a hyperplane in the feature space that maximizes the margin between classes for linearly separable data.

Random Forest:

Base Model: A single decision tree within the random forest ensemble, without any feature subsampling or tree depth limitations.

Description: Ensemble learning method that constructs multiple decision trees and aggregates their predictions to improve generalization and reduce overfitting.

Neural Networks:

Base Model: A simple feedforward neural network with a single hidden layer and a sigmoid or ReLU activation function.

Description: Comprises interconnected neurons organized in layers, where each neuron computes a weighted sum of inputs and passes it through an activation function to generate an output.

Base models provide a starting point for exploring machine learning algorithms and understanding their fundamental principles. From these base models, practitioners can then iteratively refine and optimize their models through techniques such as feature engineering, hyperparameter tuning, and ensemble learning to improve performance on specific tasks.

Model Evaluation:

Model	Accuracy
Logistic Regression (Using ROS)	90%
Gradient Boosting Classifier	82 %
Random Forest Classifier	90 %
Naïve Bayes (Multinomial NB)	81%
Support Vector Machine	93%

- Model evaluation involves assessing the performance of the trained model on unseen data.
- Common evaluation metrics depend on the type of problem (e.g., accuracy, precision, recall, F1-score for classification; mean squared error, R-squared for regression).
- The goal is to determine how well the model generalizes to new data and identify areas for improvement.

Hyperparameter Tuning:

```
Best Parameters: {'svc__C': 10, 'svc__kernel': 'linear'}
Best Accuracy: 0.9343459770355846
Accuracy (Best Model): 0.9336151839525052
Classification Report (Best Model):
              precision    recall  f1-score   support

    negative       0.86      0.91      0.88      1116
     neutral       0.70      0.74      0.72      1221
    positive       0.98      0.96      0.97      8780

    accuracy                           0.93     11117
   macro avg       0.85      0.87      0.86     11117
weighted avg       0.94      0.93      0.93     11117
```

Hyperparameter tuning involves optimizing the hyperparameters of the machine learning algorithm to improve its performance.

- Hyperparameters are settings that are not learned during training and can significantly affect the model's performance.
- Techniques for hyperparameter tuning include grid search, random search, and Bayesian optimization.

FINAL MODEL

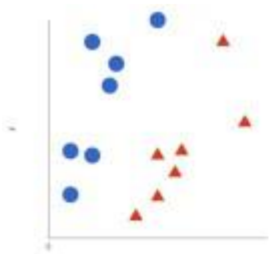
The final model is the trained machine learning model with optimized hyperparameters.

- It is selected based on its performance on the evaluation metrics and is ready for deployment in real-world applications.
- The final model represents the best solution to the machine learning problem given the available data and resources.

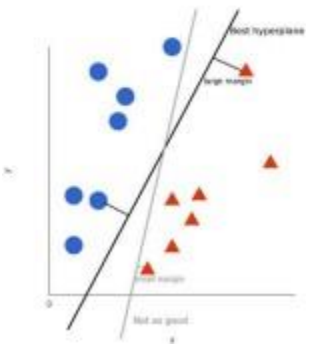
Each step in the machine learning pipeline plays a crucial role in developing accurate and reliable predictive models for various applications.

Final Model

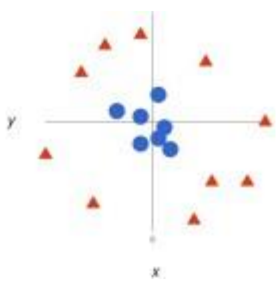
Support Vector Machines (SVM) A support vector machine is another supervised machine learning model, similar to linear regression but more advanced. SVM uses algorithms to train and classify text within our sentiment polarity model, taking it a step beyond X/Y prediction. For a simple visual explanation, we'll use two tags: red and blue, with two data features: X and Y. We'll train our classifier to output an X/Y coordinate as either red or blue.



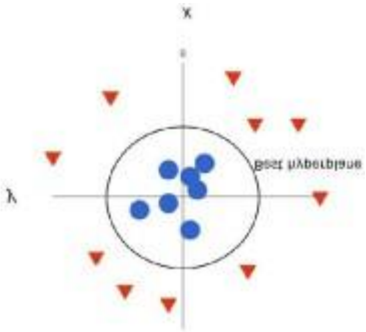
The SVM then assigns a hyperplane that best separates the tags. In two dimensions this is simply a line (like in linear regression). Anything on one side of the line is red and anything on the other side is blue. For sentiment analysis this would be positive and negative. In order to maximize machine learning, the best hyperplane is the one with the largest distance between each tag:



However, as data sets become more complex, it may not be possible to draw a single line to classify the data into two camps:



Using SVM, the more complex the data, the more accurate the predictor will become. Imagine the above in three dimensions, with a Z axis added, so it becomes a circle. Mapped back to two dimensions with the best hyperplane, it looks like this:



Very simply put, SVM allows for more accurate machine learning because it's multidimensional.

```
Best Parameters: {'svc__C': 10, 'svc__kernel': 'linear'}
Best Accuracy: 0.9343459770355846
Accuracy (Best Model): 0.9336151839525052
Classification Report (Best Model):
              precision    recall  f1-score   support

    negative       0.86      0.91      0.88      1116
     neutral       0.70      0.74      0.72      1221
    positive       0.98      0.96      0.97      8780

    accuracy                           0.93     11117
   macro avg       0.85      0.87      0.86     11117
weighted avg       0.94      0.93      0.93     11117
```

Project Justification

The sentiment analysis project on Amazon reviews aims to analyze and understand the sentiments expressed by customers in their product reviews on the Amazon platform. By employing natural language processing (NLP) techniques, the project will extract valuable insights from textual data, categorizing sentiments into positive, negative, or neutral. The primary goal is to provide Amazon and stakeholders with actionable insights that can enhance customer satisfaction, improve product offerings, and contribute to a positive online shopping experience.

• Complexity Involved:

o Large and Diverse Dataset:

▪ The project involves dealing with a large and diverse dataset comprising reviews from various product categories. Handling the complexity of diverse language use and product-specific nuances is a key challenge.

o Sentiment Ambiguity:

▪ Sentiment analysis often encounters ambiguous expressions, sarcasm, or nuanced sentiments. Developing a model that can accurately capture such nuances adds complexity to the project.

• Integration with Business Context:

▪ Ensuring that the sentiment analysis aligns with the business context of Amazon and translates into actionable strategies requires a deep understanding of the e-commerce industry.

• Project Outcome: o Commercial Value:

• Customer Experience Enhancement: The project will provide insights into customer sentiments, helping Amazon enhance product offerings, address pain points, and optimize the overall customer experience.

• Marketplace Competitiveness: Actionable insights derived from sentiment analysis can contribute to Amazon's competitive advantage by identifying areas for improvement and innovation.

o Academic Value:

• Research Contribution: The project can contribute to academic research in the fields of natural language processing, sentiment analysis, and e-commerce analytics. Findings and methodologies can be valuable for future research endeavors.

o Social Value:

• Consumer Empowerment: By understanding customer sentiments, the project contributes to empowering consumers. It helps potential buyers make informed decisions by providing insights into the experiences of previous purchasers.

• Brand Reputation: Positive sentiment analysis outcomes contribute to building and maintaining a positive brand reputation for Amazon, reinforcing trust among customers.

• Overall Impact:

The sentiment analysis project on Amazon reviews aims to use advanced language analysis techniques to understand and categorize customer sentiments. This analysis will provide valuable insights for Amazon to enhance the online shopping experience, improve services, and address customer concerns. Beyond its commercial impact, the project also contributes to academic research in the fields of sentiment analysis and ecommerce analytics, benefiting a wider community interested in understanding customer opinions and behavior.

DRAWBACKS

Sentiment analysis, like any other natural language processing (NLP) technique, comes with its own set of drawbacks and limitations. Here are some of the key drawbacks of sentiment analysis:

Ambiguity and Context Dependency: Sentiment analysis often struggles with understanding context and nuances in language. Words or phrases may carry different sentiments depending on the context in which they are used. For example, "This movie is sick" could be interpreted as positive by one person and negative by another, depending on their interpretation of the slang term "sick."

Sarcasm and Irony: Sentiment analysis models may have difficulty identifying sarcasm and irony in text. These forms of language often involve saying the opposite of what is meant, leading to misclassification of sentiment. For example, "Great job, Einstein" may appear positive on the surface but is actually sarcastic.

Negation and Modifiers: Negation words (e.g., not, never) and modifiers (e.g., very, extremely) can drastically alter the sentiment of a statement. Sentiment analysis models may struggle to accurately account for these linguistic nuances, leading to misinterpretation of sentiment.

Subjectivity and Opinion Variability: Sentiment is inherently subjective and can vary greatly among individuals. What one person considers positive, another might perceive as negative. Sentiment analysis models trained on one dataset may not generalize well to new or diverse datasets due to differences in opinion and expression.

Domain Specificity: Sentiment analysis models trained on general-purpose datasets may not perform well in domain-specific contexts. The language and sentiment expressions used in domains such as healthcare, finance, or legal may differ significantly from those in everyday conversation or social media.

Data Imbalance: Sentiment analysis datasets often suffer from class imbalance, where one sentiment class (e.g., positive) may be significantly more prevalent than others (e.g., negative or neutral). This imbalance can bias the model's predictions and affect its overall performance.

Language and Cultural Bias: Sentiment analysis models trained on data from specific languages or cultural contexts may exhibit bias when applied to text from different languages or cultures. Cultural differences in language use and expression can impact the accuracy and reliability of sentiment analysis results.

Data Noise and Sparsity: Text data used for sentiment analysis often contain noise, such as spelling errors, grammatical mistakes, slang, and abbreviations. Additionally, sentiment analysis models may struggle with sparse data, especially for less common sentiment expressions or languages.

Addressing these drawbacks often requires a combination of techniques, including fine-tuning models on domain-specific data, incorporating context-awareness into algorithms, and utilizing ensemble methods to improve robustness and generalization. Despite these limitations, sentiment analysis remains a valuable tool for analyzing and understanding public opinion, customer feedback, and textual data at scale.

CONCLUSION

In this study, we proposed sentiment diffusion for customer review analysis. Suggested sentiment dissemination increased the consistency of existing sentiment analysis domains by reinforcing sentiment words in both context- and domain-specific ways. With the help of sentiment analysis implementation, we can conclude that most people are satisfied with the products and services they received from Amazon. Sentiment analysis is a good process to gather insights into each product, which ultimately benefit present and future customers and e-commerce companies. Sentiment analysis is important because, based on bad reviews, the e-commerce company makes those products better or replaces those products with better and newer ones, which ultimately improves the overall customer service too.

Source

- What is Retail Analytics? | Sisense Sentiment Analysis & Machine Learning (monkeylearn.com)
- https://www.kaggle.com/code/amuhialdeen/lstm-in-amazon-customer-sentiment