**Sentiment Analysis of Amazon Product Review data using VADER**

Online marketplaces ask for customer feedback to improve their products and marketing strategies. However, customers may find conflicting reviews and opinions challenging to navigate (Analyzing reviews is critical for creating effective marketing tactics. Sentiment analysis determines user attitudes toward a product .This report uses VADER ( Valence Aware Dictionary for Sentiment Reasoning) to analyze Amazon reviews of three products, presenting results using graphs and word clouds.

# This section outlines the implementation methodology. The first part covers coding environments, while the second part discusses the pre-processing steps and proposed system. The Vader analyzer process and its implementation are shown in the last section.

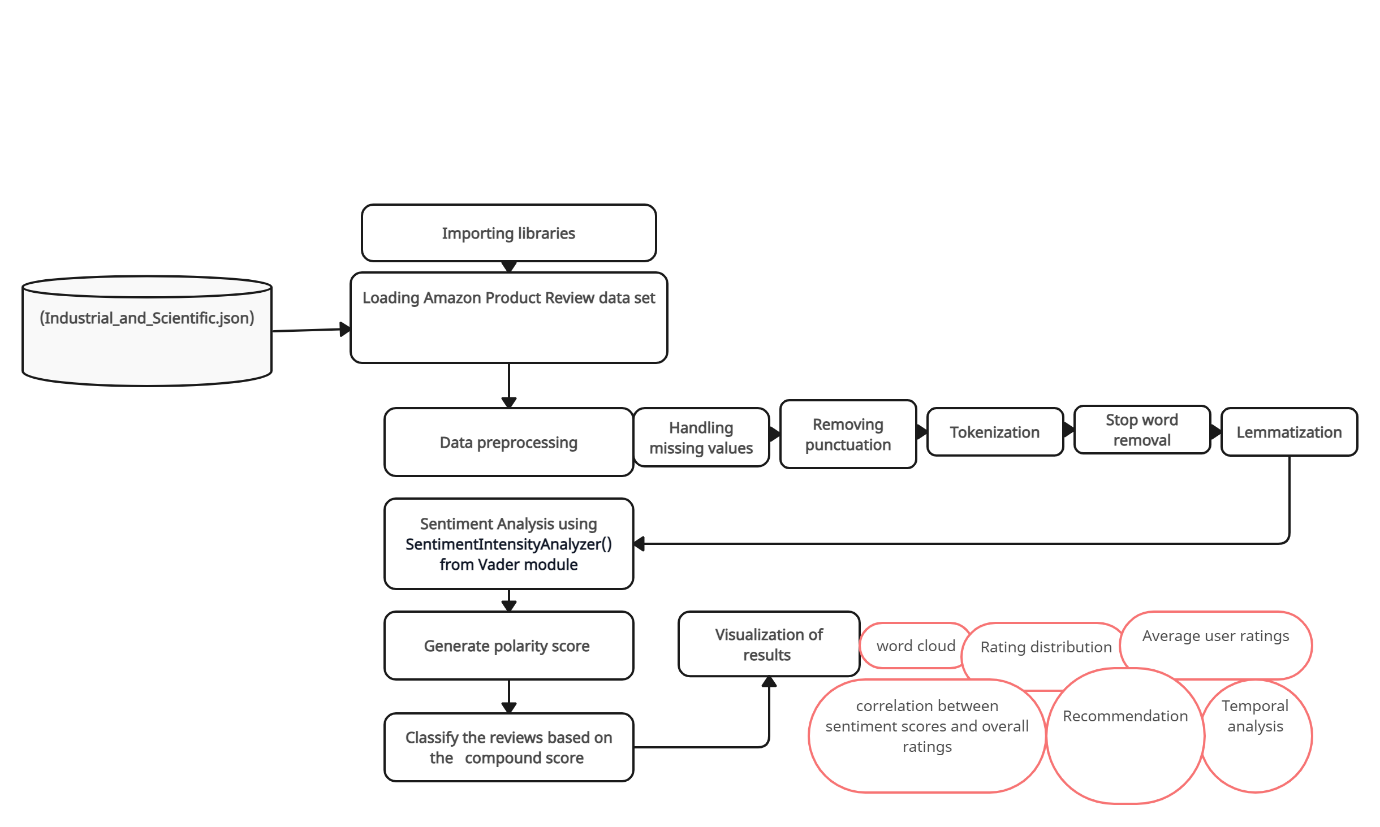


Figure 2.1 Architectural Diagram

VADER sentiment analyzer tool in Python requires NLTK library for analyzing linguistic data. NLTK provides crucial modules for preprocessing and analysis, as shown in Figure 2.1 of the proposed method's architecture. These modules are imported easily from NLTK, as shown in figure 2.2.

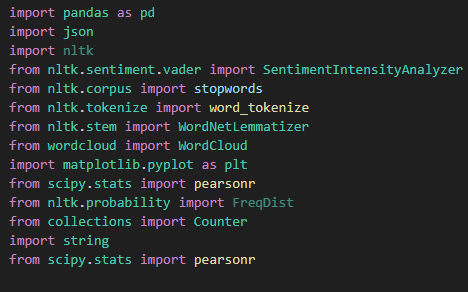


Figure 2.2

Step two involves loading a JSON data file, converting it to Python dictionaries using json.loads, and saving a specific product into a data frame using the 'asin' value. The data frame is then cleaned by handling missing and NaN values (Figure 2.3).

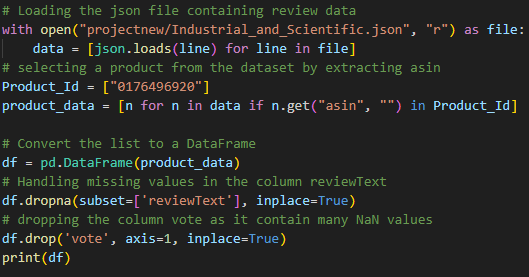


Figure 2.3

Pre-processing of review Text

nltk sentiment.vader: This module offers the sentiment analysis tool VADER, which classifies text into three categories: neutral, negative, and positive feelings.

nltk.corpus This module is used for accessing the NLTK corpora, which includes datasets and resources for various natural language processing tasks such as stop word removal .

nltk.tokenize: This module offers multiple tokenization techniques to separate text into words or phrases.

nltk.stem: Classes for stemming, which reduces words to their base form, are included in this module. WordNetLemmatizer is used for lemmatization.

The normalize\_text() function removes punctuation marks, hyphens, and full stops and tokenizes the text, removes the stop words and finally lemmatizes as shown in figure 2.4.

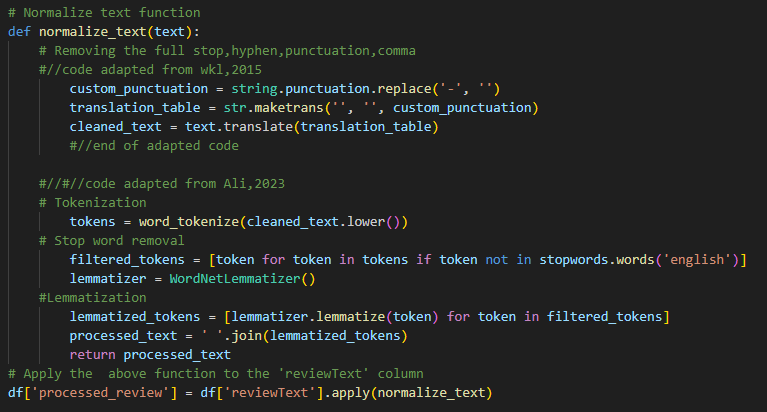


Figure 2.4

In Figure 2.5, DataFrame uses SentimentIntensityAnalyzer to evaluate the sentiment of text reviews and generates, the polarity\_score assigns a sentiment category based on compound score. Scores above 0.05 are positive, below -0.05 are negative, and anything else is neutral (Bonta et al., 2019). The sentiment is added to a new column named 'sentiment'.

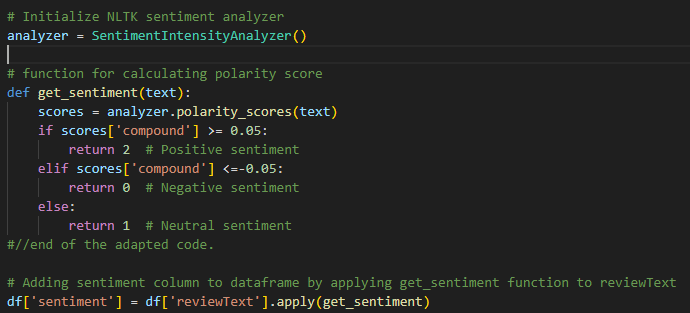


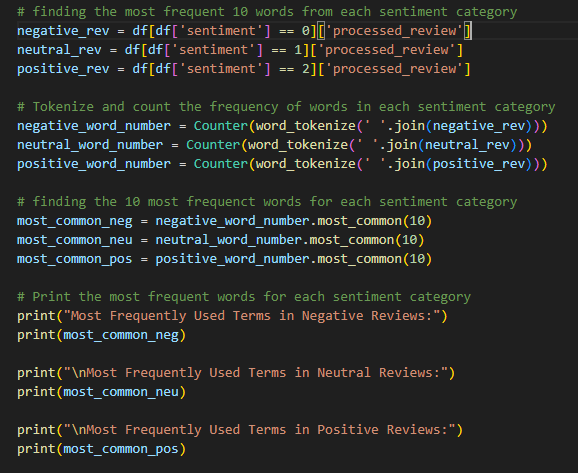
Figure 2.5 Sentiment Analyser using Vader.

After classifying the review text, the words from positive, negative, and neutral comments can be plotted in the word cloud using the code snippet below.(figure2.6)



Figure 2.6 Plotting word cloud.

The code analyzes and displays commonly used words in reviews with neutral, positive, or negative sentiments. Based on the sentiment column, the DataFrame df is divided into three subsets which contain negative reviews, neutral reviews, and positive reviews. The parsed review text is tokenized for each sentiment category using the NLTK word\_tokenize function. The Counter class counts the tokenized words by frequency, and the most\_common() method is used to get the top ten most frequently occurring words in each sentiment category. (Figure 2.7)



S

3. Results and Visualization of Analysis

Following sentiment analysis, a few rows of processed review text and sentiment labels are displayed in Figure 3.

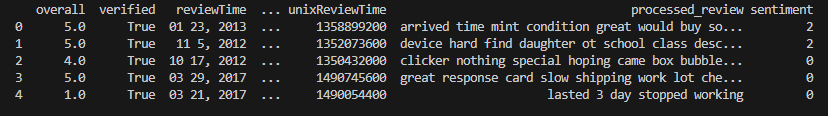


Figure 3

3.1 Word Cloud Visualization

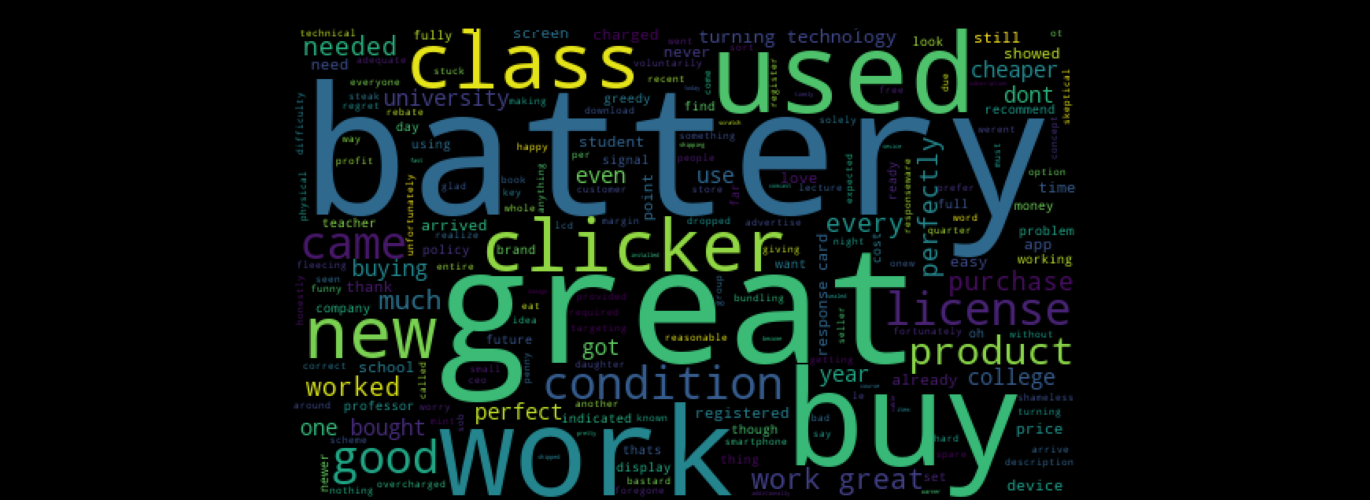


Figure 3.1.1 Positive\_Review Word Cloud



Figure 3.1.2 Neutral\_review Word cloud

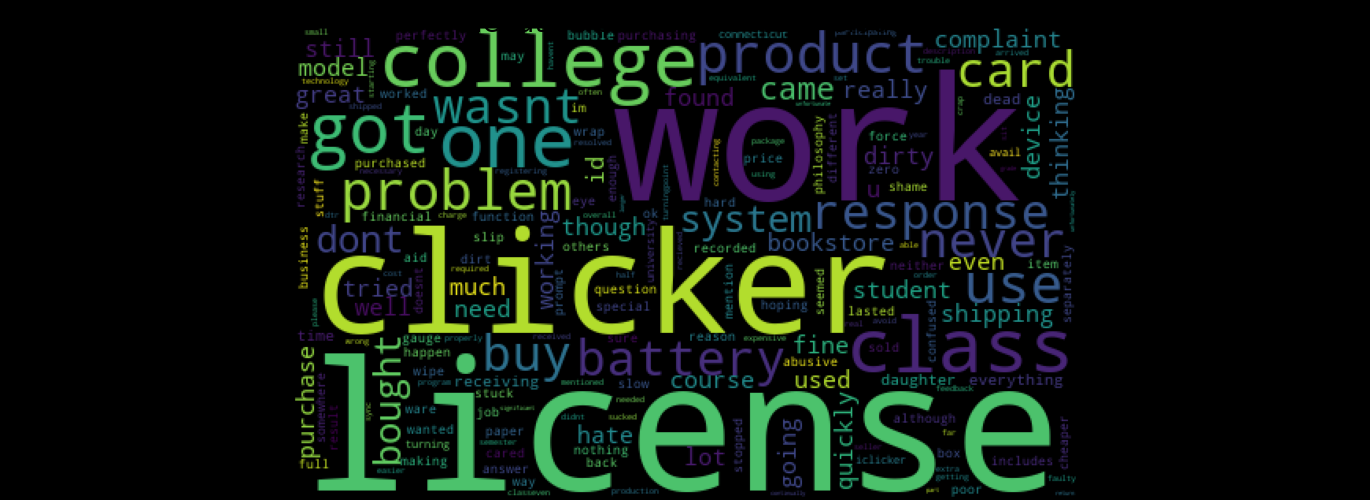


Figure 3.1.3 Negative\_review Word cloud

3.2 Distribution of sentiments over the years

To analyze product review sentiments, group the data frame by 'Year' and 'sentiment,' and calculate the size of each group using the 'size' method. Plot the count of positive, negative, and neutral reviews over time. Store the result in a data frame named 'Sentiment\_Year.' Figure 3.2.1 shows the code snippet.

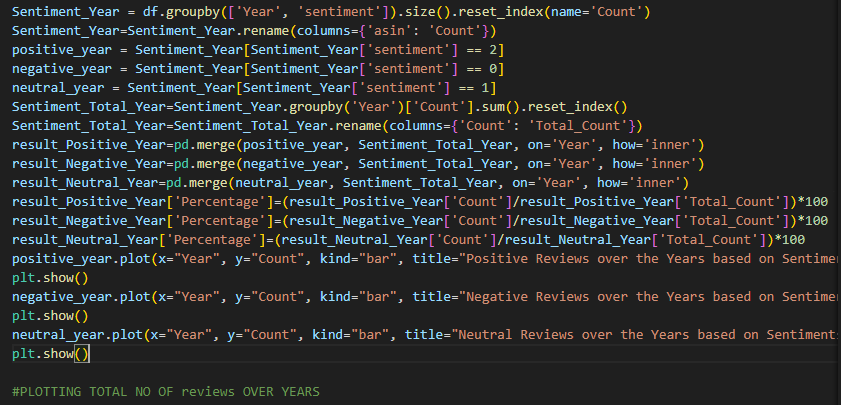


Figure 3.2.1 Plotting the distribution of sentiments.

The code mentioned above provides a temporal analysis of each sentiment category over time, as demonstrated in Figures 3.2.2 and 3.2.3. This shows a decline in positive ratings between 2015 and 2018. The number of unfavorable product reviews increased between 2014 and 2017. Conversely, figure 3.2.3 shows that neutral reviews increased in 2016 but decreased in 2017.

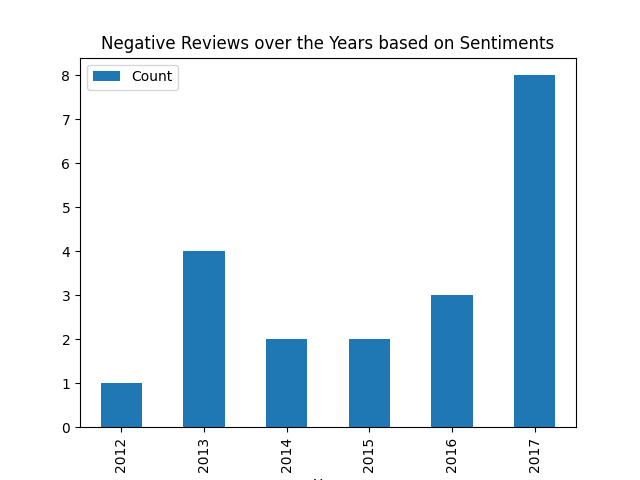
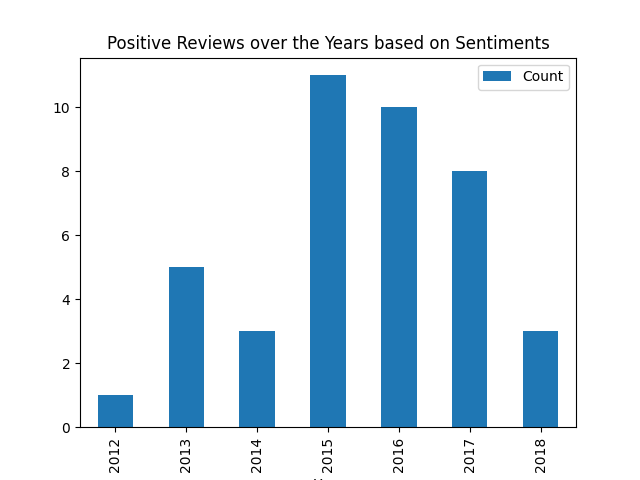


Figure 3.2.2 Positive\_reviews and negative\_reviews versus years

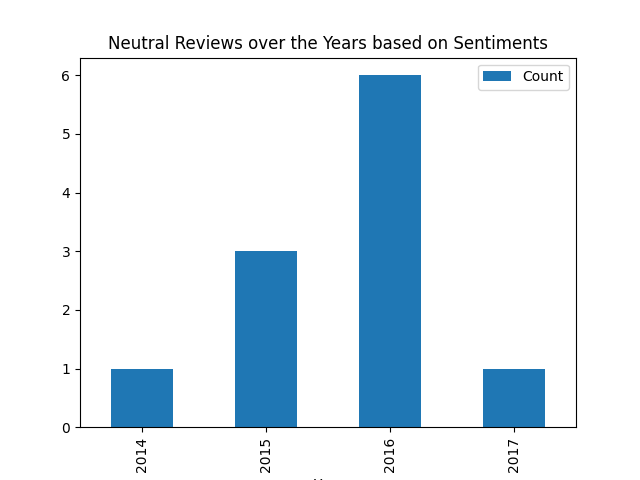


Figure 3.2.3 Neutral reviews over years.

3.3 Distribution of the total no of reviews over the years

The 'Year' column groups the DataFrame, and the 'asin' column counts to find the number of reviews each year. The result is stored and used to plot a line chart that shows the trend over time as shown in Figure 3.3.1

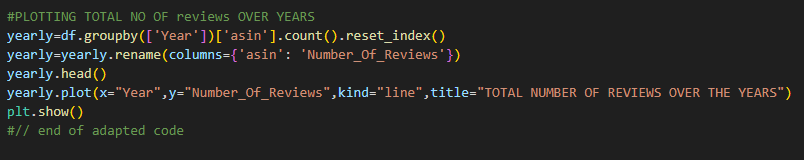


Figure 3.3.1 plot the total number of reviews over time

The line graph below indicates that from 2014 to 2016, product reviews were rising, but by 2018, they had abruptly decreased. This suggests that product demand decreased by 2018.

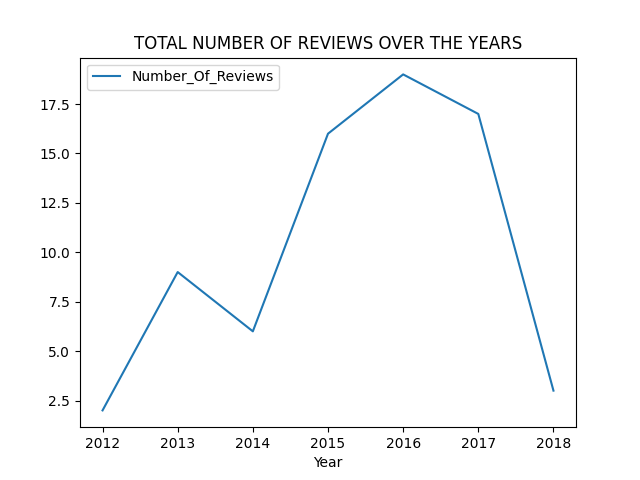


Figure 3.3.2 Line graph

3.4 Distribution of sentiment polarity scores

The code segment figure 3.4.1 creates a histogram to visualize the distribution of sentiment polarity scores in the provided data frame.

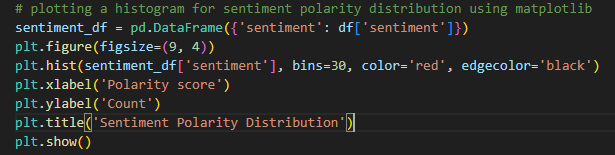


Figure 3.4.1

Figure 3.4.2 implies the count of positive reviews is high compared to other categories.

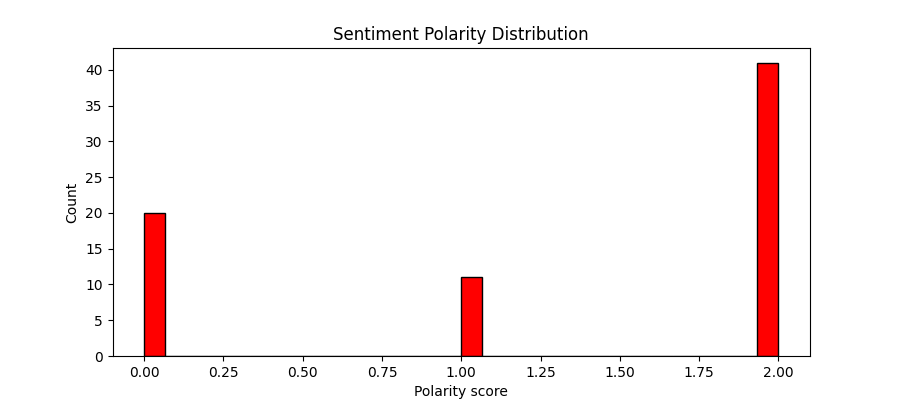


Figure 3.4.2 Polarity distribution

3.5 Distribution of Review Rating

Figure 3.5.1 generates a histogram that shows the distribution of overall review ratings. Figure 3.5.2 illustrates that the count of top rating of five stars is high.

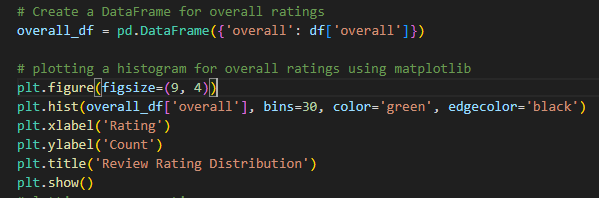


Figure 3.5.1

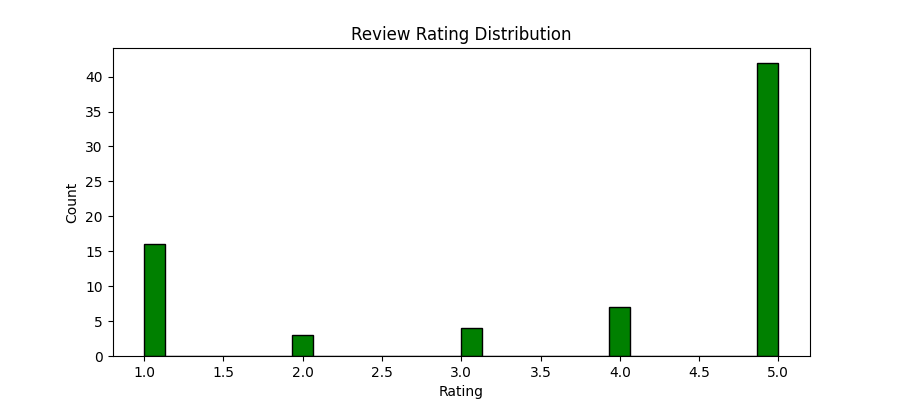


Figure 3.5.2 Distribution of rating.

3.6 Average review rating by users

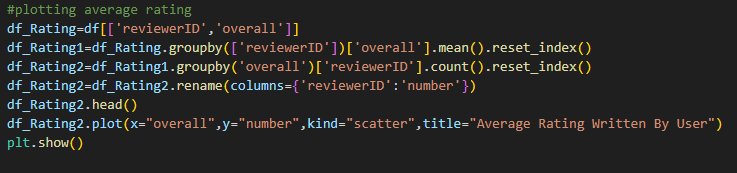


Figure 3.6.1

The above code snippet performs an analysis on the average rating given by each reviewer and plots a scatter plot figure 3.6.2 to visualize the distribution.

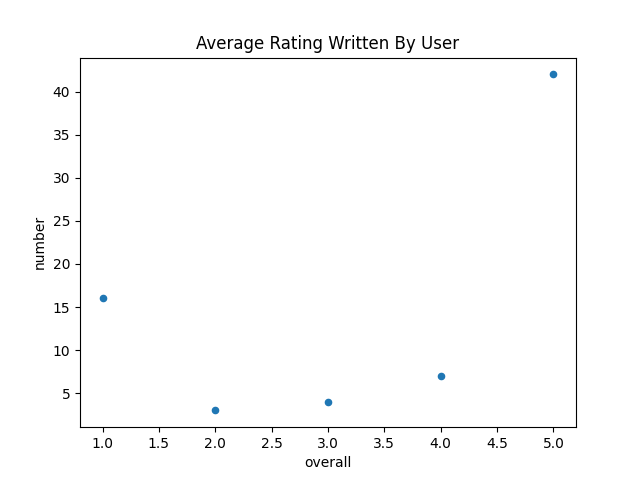


Figure 3.6.2

3.7 Calculating Pearson correlation between sentiment and overall rating

The code below calculates the Pearson correlation coefficient between the sentiment scores and overall ratings in the DataFrame.

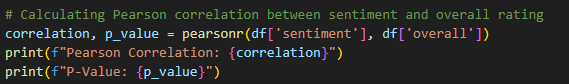


Figure 3.7.1

The two variables show a positive linear relationship with a correlation coefficient of 0.476. This means that as sentiment scores increase, overall ratings also tend to increase. The p-value is below 0.05, indicating statistical significance. The figure 3.7.2 shows the regression line representing the linear relationship between them.

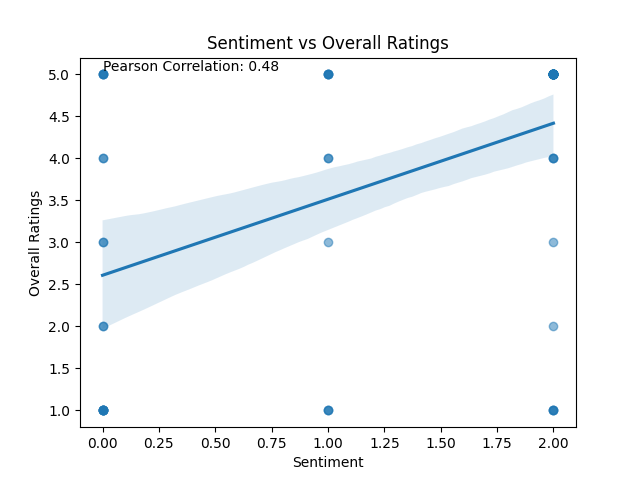


Figure 3.7.2

3.8 Most Frequently Used Words and Recommendations

Figure 3.8.1 shows the ten most frequent words in each sentiment and the generated recommendations to the manufacturer.

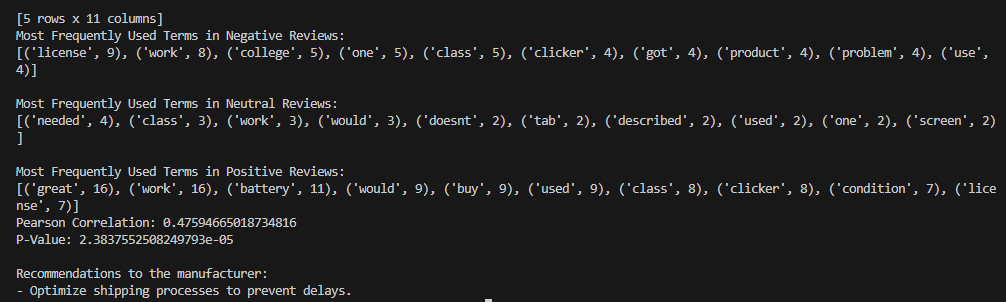


Figure 3.8.1

The code snippet 3.8.2 generates the recommendations, and it is crucial for future developments. These ideas allow the business to modify its tactics and offerings.

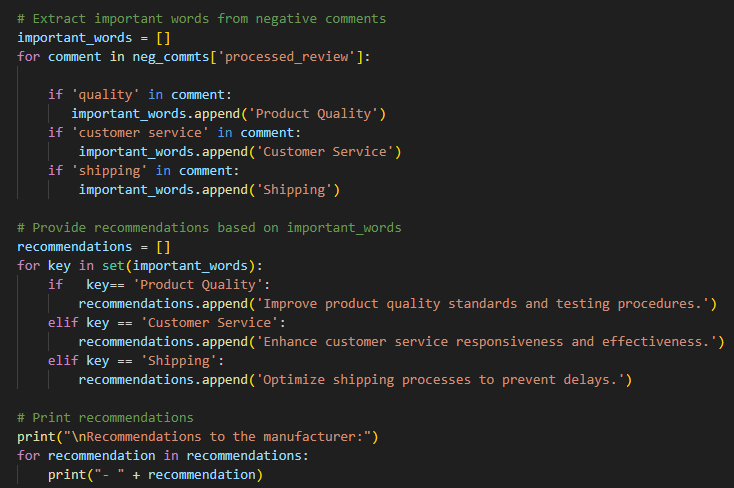


Figure 3.8.2

Two more products using the same methodology and code are in the appendix.

# CONCLUSION

To summarize, VADER sentiment analysis has made extracting sentiment from product evaluation data easier. It provides insightful and intricate sentiment ratings for each review in the dataset, making it effective for social media and short-text sentiment analysis. Although the dataset is unlabeled, VADER performs well with this data as I discovered during manual evaluation of a few random review texts.