A Real time research project report

On

# Enhancing Reliability Prediction in Amazon Reviews

submitted in partial fulfillment of the requirements for the award of degree of

**BACHELOR OF TECHNOLOGY**

in

**COMPUTER SCIENCE & ENGINEERING**

(Artificial Intelligence & Machine Learning)

by


**23WH1A6606 Ms. DANDU AKSHAYA**

**23WH1A6627 Ms. G HEMA ASHRITA**

**23WH1A6630 Ms. MUDUNDI SHANMUKHI**

**24WH5A6603 Ms. KODUMAGULLA THRIPADA**

**under the esteemed guidance of**

**Ms. Vuyyuru Asha**

**Assistant Professor, CSE(AI&ML)**

**BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN**

**(UGC Autonomous Institution | Approved by AICTE | Affiliated to JNTUH)**

**(NAAC Accredited - A Grade | NBA Accredited B.Tech. (EEE, ECE, CSE and IT)**

**Bachupally, Hyderabad – 500090**


**April-2025**

# BVRIT HYDERABAD
# COLLEGE OF ENGINEERING FOR WOMEN

**(UGC Autonomous Institution | Approved by AICTE | Affiliated to JNTUH)**

**(NAAC Accredited - A Grade | NBA Accredited B.Tech. (EEE, ECE, CSE and IT)**

**Bachupally, Hyderabad – 500090**

## Department of Computer Science & Engineering
## (Artificial Intelligence & Machine Learning)

## CERTIFICATE

This is to certify that the Real time research project entitled **"ENHANCING RELIABILITY PREDICTION IN AMAZON REVIEWS"** is a bonafide work carried out by **Ms. D. Akshaya (23WH1A6606), Ms. G. Hema Ashrita (23WH1A6627)**, **Ms. M. Shanmukhi (23WH1A6630), Ms. K. Thripada (24WH5A6603**) in partial fulfillment for the award of B.Tech degree in **Computer Science & Engineering (AI&ML)**, **BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad**

| | |
|---|---|
| **Internal Guide** | **Head of the Department** |
| **Ms. Vuyyuru Asha** | **Dr. B. Lakshmi Praveena** |
| **Assistant Professor** | **HoD & Professor** |
| **Dept of CSE(AI&ML)** | **Dept of CSE(AI&ML)** |

# DECLARATION

We hereby declare that the work presented in this project entitled **“ENHANCING RELIABILITY PREDICTION IN AMAZON REVIEWS”** submitted towards completion of real time research project work in II Year of B.Tech of CSE(AI&ML) at **BVRIT HYDERABAD College of Engineering for Women,** Hyderabad is an authentic record of our original work carried out under the guidance of **Ms. Vuyyuru Asha, Assistant Professor, Department of CSE(AI&ML).**

**Sign with Date:**

Ms. D. Akshaya

(23WH1A6606)

**Sign with Date:**

Ms. G. Hema Ashrita

(23WH1A6627)

**Sign with Date:**

Ms. M. Shanmukhi

(23WH1A6630)

**Sign with Date:**

Ms. K. Thripada

(24WH5A6603)

# ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. K. V. N. Sunitha, Principal, BVRIT HYDERABAD College of Engineering for Women**, for her support by providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. B. Lakshmi Praveena, Head of the Department, Department of CSE (AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for all timely support and valuable suggestions during the period of our project.

We are extremely thankful to our Internal Guide, **Ms. Vuyyuru Asha, Assistant Professor, CSE (AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for her constant guidance and encouragement throughout the project.

We would like to thank our Real Time Project Coordinators, **Mr. K. Sundeep Saradhi, Assistant Professor, CSE (AI&ML), BVRIT HYDERABAD College of Engineering for Women,** and all the Staff of Department of CSE (AI&ML) department who helped us directly or indirectly through out the project.

Finally, Last but not least, we wish to acknowledge our **Parents** and **Friends** for giving moral strength and constant encouragement.

**Ms. D Akshaya**

**(23WH1A6606)**

**Ms. G. Hema Ashrita**

**(23WH1A6627)**

**Ms. M. Shanmukhi**

**(23WH1A6630)**

**Ms. K. Thripada**

**(24WH5A6603)**

# Abstract

In this project, we aimed to build developed an automated web scraping system using Selenium to extract customer review data from Amazon. The scraper efficiently gathered key information, including the review content, rating, helpful vote count, verified purchase status, and reviewer name. This data collection focused on building a comprehensive dataset of customer sentiments associated with various products.

Once collected, the review data underwent sentiment analysis using a natural language processing approach. Each review was analyzed to determine its sentiment category—positive, negative, or neutral—along with corresponding sentiment scores and word counts. This helped quantify the emotional tone of customer feedback across a diverse range of reviews.

The dataset was then preprocessed for accuracy and relevance. Reviews with fewer than ten words and those classified as neutral were removed to enhance the quality of insights. This cleaning step ensured that only informative and emotionally charged reviews were used in the analysis.

Finally, Exploratory Data Analysis (EDA) was conducted to visualize trends and patterns in customer sentiments. The analysis revealed meaningful insights into customer satisfaction and product perception, laying the groundwork for future applications such as product improvement, customer experience enhancement, and review- based recommendations system

# INDEX

# INTRODUCTION:

In today's digital world, online reviews play a crucial role in shaping customer decisions and influencing product success. However, the vast volume of user-generated content can make it difficult for businesses and customers to interpret opinions accurately. This project aims to address that issue by collecting and analysing Amazon product reviews using automated scraping and sentiment analysis, offering a clearer picture of customer feedback.

Using Selenium, reviews were scraped directly from Amazon, capturing key details such as the review text, rating, reviewer name, helpfulness votes, and whether the purchase was verified. This raw data was then cleaned and filtered—removing short and neutral reviews—to focus on more meaningful opinions. This helps in reducing noise and improving the accuracy of sentiment interpretation.

The processed data was then analysed through sentiment analysis, classifying reviews as positive, negative, or neutral, and computing their respective sentiment scores. Exploratory Data Analysis (EDA) was performed to uncover trends, word usage patterns, and the overall sentiment distribution. This not only helps businesses understand customer perceptions but also enables data-driven decision-making for product improvement and marketing strategies.

## 1.1. Objectives

The main objectives of this project are as follows:

1. **Automate Data Collection**
   Develop an automated data collection system utilizing web scraping techniques to gather reviews from Amazon website.

2. **Feature Extraction**
   Extract key property features including the Name, Review, Review Text, Helpful votes for the review, and valid purchase.

3. **Data Organization and Storage**
   Structure and store the extracted data in a standardized format (CSV) to facilitate seamless analysis and modeling.

4. **Exploratory Data Analysis**
   Exploratory Data Analysis (EDA) was performed to identify sentiment trends, frequent word patterns, and overall customer opinion distribution across the collected Amazon reviews.

5. **Model Development**
   A sentiment analysis model was developed using NLP techniques to classify Amazon reviews into positive, negative, and neutral categories based on their textual content.

6. **Model Evaluation**
   The model was evaluated using accuracy, precision, recall, and F1-score to assess its performance in correctly classifying the sentiment of Amazon reviews.

7. **System Usability and Insights**
   The system demonstrated strong usability by automating data collection and sentiment classification efficiently, providing valuable insights into customer opinions, such as common concerns, product strengths, and overall satisfaction trends.

## 1.2. Existing System

- **MonkeyLearn** is a user-friendly, no-code platform designed to perform text analysis tasks such as sentiment analysis, keyword extraction, and topic classification. It is widely used by businesses to analyze customer feedback, reviews, and support tickets, helping them understand customer sentiment at scale. With an intuitive interface and ready-to-use models, it enables quick insights without requiring extensive programming knowledge.

- One of its key features is the ability to analyze text data in real-time and generate sentiment labels along with confidence scores. Users can either use pre-trained models or create custom ones by training on their own datasets. This makes MonkeyLearn accessible to both non-technical users and those who want more control over their analysis pipeline.

- However, despite its advantages, MonkeyLearn has some limitations. The pre-trained models may not capture context-specific language or industry-specific nuances, leading to potential inaccuracies in certain use cases. Additionally, advanced customization and integration options are often locked behind premium plans, which may not be ideal for small-scale projects or academic use.

Overall, the existing system does not leverage modern technology like machine learning and data analytics to assist users in decision-making.

## 1.3. Proposed System

The proposed system introduces a **technology-driven solution** that automates data gathering and analysis.

The proposed system is a custom-built sentiment analysis pipeline designed to extract and analyze Amazon product reviews. It uses Selenium for web scraping to collect detailed review data including the review content, rating, helpfulness count, reviewer name, and purchase verification. This raw data is then cleaned and filtered to remove neutral or very short reviews, ensuring only meaningful feedback is used for analysis.

The system applies Natural Language Processing techniques to perform sentiment analysis, categorizing reviews into positive, negative, or neutral classes and assigning sentiment scores. This is followed by exploratory data analysis (EDA) to uncover trends, commonly used words, and sentiment distribution. The model provides deeper, more tailored insights than generic tools, as it focuses specifically on product-related reviews and includes custom pre-processing steps to improve result accuracy.

Compared to existing solutions, the proposed system offers several advantages: full control over the scraping and analysis process, customization to specific domains or product types, better transparency in how the sentiment model works, and flexibility to refine or extend features as needed. This makes it ideal for businesses or researchers looking for more precise, actionable insights from customer reviews.

## 2. Literature Survey

---

The rise of online product reviews has significantly influenced consumer behavior, leading to an increased focus on sentiment analysis and text mining. Below are key points from existing literature that are relevant to your project:

1. Importance of Sentiment Analysis:
   - Sentiment analysis, also known as opinion mining, is a subfield of Natural Language Processing (NLP) focused on determining the sentiment expressed in a given text (Pang & Lee, 2008).
   - Sentiment analysis is increasingly used to understand consumer opinions in reviews, guiding businesses in product development, marketing strategies, and customer service.

2. Challenges with General Sentiment Analysis Models:
   - While many commercial tools like MonkeyLearn and Lexalytics provide sentiment classification, they often struggle with domain-specific contexts (Liu, 2012).
   - Generalized sentiment models, especially those trained on generic data, may fail to capture the nuances of product-specific language. For instance, a "positive" sentiment on a product review might be context-dependent, depending on the features being discussed.
   - In addition, such tools often do not allow fine-tuned customization for specific product categories or detailed filtering of data (e.g., removing neutral or irrelevant reviews).

3. The Need for Custom Solutions:
   - Web scraping has become an essential technique for gathering real-time review data directly from online platforms like Amazon (Gopi & Kumar, 2020). Tools like Selenium are commonly used to extract detailed information, including reviewer names, ratings, helpful votes, and purchase validation.
   - By using Selenium for scraping, researchers can access granular data that is not available via public APIs or standardized review aggregation tools. This ensures that critical metadata is captured for further analysis.

4. Sentiment Analysis Models and Their Application:
   - Various sentiment analysis models have been used for analyzing product reviews. One common approach is the use of VADER (Valence Aware Dictionary and sEntiment Reasoner), which is highly effective for social media and online reviews because it accounts for punctuation, capitalization, and slang (Hutto & Gilbert, 2014).
   - Machine learning-based approaches, including supervised learning algorithms (e.g., Naive Bayes, SVM, and logistic regression), are also employed to classify reviews into positive, negative, or neutral categories based on labeled training data (Medhat et al., 2014).

5. Data Cleaning and Preprocessing for Improved Accuracy:
    - A major step in ensuring the quality of sentiment analysis is data preprocessing. This involves cleaning the data to remove irrelevant or noisy information, such as short or neutral reviews (Medhat et al., 2014).
    - Removing neutral and very short sentences is crucial, as these types of reviews do not provide strong emotional signals, which could impact the overall sentiment analysis accuracy.
    - Additionally, reviews with a verified purchase status are typically given more weight, as they tend to be more reliable and trustworthy.

6. Exploratory Data Analysis (EDA) in Sentiment Analysis:
    - EDA is a critical step in understanding the data distribution and uncovering patterns in sentiment. After sentiment classification, EDA is used to identify trends, common word usage, sentiment distribution, and relationships between helpful votes and sentiment.
    - Visualizations, such as word clouds, frequency plots, and sentiment trend graphs, help to provide a clear overview of customer opinions (Tufte, 2001).

7. Advantages of Custom-Built Systems:
    - Custom-built sentiment analysis systems provide several advantages over existing commercial solutions. They offer:
        - Flexibility: Tailored specifically to the product domain or research needs, allowing for a more accurate analysis.
        - Transparency: Full control over the scraping and sentiment classification process, ensuring that the data handling and model choices are fully understood and customizable.
        - Customization: Ability to fine-tune pre-processing steps like removing neutral reviews or short sentences, which can significantly impact the quality of insights.
        - Scalability: Custom systems can scale according to data collection and analysis needs, supporting a broader range of review sources and deeper analysis.

8. Limitations of Existing Solutions:
    - MonkeyLearn and other sentiment analysis platforms often lack deep customization options. These tools rely on pre-trained models, which may not accurately capture the nuances of product-specific sentiment (Liu, 2012).
    - In some cases, these systems lack access to raw data, limiting the potential for detailed analysis and customization (e.g., granular filtering or advanced pre-processing steps).
    - Data limitations: Many commercial tools may also struggle to handle large-scale or unstructured data from diverse sources, often requiring manual intervention or additional processing steps to maintain accuracy.

# 3. System Requirements

## 3.1 Software Requirements:

- Operating System : Windows 10 / macOS Catalina
- Python : Python 3.8 or higher
- Google Chrome : Up-to date version(for Scraping)
- ChromeDriver : Compatible with Chrome
- Together.ai : LLM API
- Text Editor/IDE : VS Code / PyCharm/ Jupyter Notebook

## 3.2 Hardware Requirements:

- RAM : 16 GB or more
- Storage : 50 GB SSD
- Display : 1080p or higher
- Network : Stable Internet Connection

## 3.3 Python Packages / Libraries:

- selenium
- pandas
- scikit-learn
- xgboost
- joblib
- spacy
- requests
- matplotlib / seaborn (for visualization)
- en_core_web_sm (for spaCy)

# 4. Methodology

The methodology section outlines the step-by-step process followed in this project, starting from data collection to final model testing and evaluation. It details the tools, technologies, and techniques used to build prediction model using web scraping and machine learning.

## 4.1. System Architecture

The system is designed to automate the process of scraping reviews, performing sentiment analysis, and visualizing insights. The architecture consists of the following components:

- **Data Collection (Web Scraping)**: Using **Selenium**, the system scrapes detailed product reviews from Amazon, including key information such as the review text, rating, helpfulness votes, reviewer name, and purchase verification status.
- **Data Preprocessing**: Raw data is cleaned to remove neutral and very short reviews, ensuring that only meaningful, informative reviews are retained for analysis.
- **Sentiment Analysis Model**: The cleaned reviews are then fed into a sentiment analysis model that classifies reviews into **positive**, **negative**, or **neutral** categories based on the content of the review. The model assigns sentiment scores to each review.
- **Exploratory Data Analysis (EDA)**: Visualizations and statistical summaries are generated to reveal insights from the sentiment analysis, such as sentiment distribution, word frequency, and trends.
- **User Interface/Results Display**: The results are presented to the user in an accessible format, such as graphs, charts, and summarized insights, allowing users to interpret the data effectively.

The entire process is automated, ensuring scalability and efficiency in handling large volumes of reviews.

## 4.2. Datasets

The dataset used for this project is sourced from **Amazon** product reviews. The key steps for obtaining the dataset are:

- **Review Extraction**: Using Selenium, the system scrapes reviews from Amazon product pages. For each product, the reviews include the following attributes:
  - **Review Text**: The content of the customer's feedback.
  - **Rating**: A numerical rating (1-5 stars) associated with the review.
  - **Helpful Votes**: The number of users who found the review helpful.
  - **Purchase Verification**: Whether the review was made by a verified purchaser.
  - **Reviewer Name**: The name or ID of the reviewer (if available).
- **Data Filtering**: The system filters out reviews that are:
  - Less than 10 words long (to remove uninformative content).
  - Neutral in sentiment, as these do not contribute to the analysis.

The dataset is dynamically updated through web scraping, ensuring that the data remains current and relevant.

## 4.3. Implementation

The implementation of the system follows these key steps:

1. **Web Scraping with Selenium**:
   - **Selenium WebDriver** is used to automate the process of navigating Amazon product pages and extracting reviews. The scraper is configured to handle pagination, ensuring it collects reviews from multiple pages.
   - **Data Extraction**: Specific fields like review text, rating, helpfulness votes, and verification status are extracted and stored in a structured format (e.g., CSV, JSON).

2. **Data Preprocessing**:
   - **Cleaning**: The review text is processed to remove stopwords, punctuation, and irrelevant characters using **NLTK** or **spaCy** libraries. Short reviews and neutral sentences are filtered out to ensure that only meaningful reviews are used.
   - **Tokenization and Lemmatization**: The text is tokenized (split into individual words) and lemmatized (reducing words to their base form) to improve the accuracy of sentiment analysis.

3. **Sentiment Analysis**:
   - **LLM-Based Sentiment Analysis**: A Large Language Model (LLM) is used via API to classify customer reviews into **positive**, **negative**, or **neutral** categories. This model provides **context-aware**, **nuanced sentiment scores** (ranging from -1 to 1), enabling more accurate interpretations of sentiment, especially in **ambiguous or complex sentences**.
   - **Machine Learning Classifier**: Optionally, a **supervised machine learning model** (e.g., Naive Bayes or SVM) can be trained on labelled data to refine sentiment classification and improve model performance.

4. **Exploratory Data Analysis (EDA)**:
   - **Visualization**: Tools like **Matplotlib** and **Seaborn** are used to generate visualizations such as bar charts, word clouds, and sentiment distribution graphs to highlight patterns in the data.
   - **Trend Analysis**: Temporal trends and sentiment correlations with helpfulness votes or ratings are analysed.

5. **Results Display**:
   - The results are presented to the user in an easy-to-read format, including sentiment scores, word frequency analysis, and visual graphs summarizing the findings.

## 4.4. Testing

The system is rigorously tested in multiple stages to ensure it functions as intended:

1. **Unit Testing**:
   - Individual components, such as the web scraper, sentiment analysis model, and data preprocessing steps, are tested independently for correctness and robustness.
   - Edge cases, such as empty reviews, missing data, or incorrectly formatted reviews, are handled to ensure the system does not break.

2. **Integration Testing**:
   - After unit testing, the system components are integrated and tested together. This ensures that the entire pipeline (from scraping to sentiment analysis and results display) works smoothly.
   - Data flows are verified from extraction through to final presentation, and performance issues are addressed if necessary.

3. **Accuracy Evaluation of Sentiment Analysis**:
   - The sentiment analysis model is evaluated using standard metrics such as **accuracy**, **precision**, **recall**, and **F1-score**. This helps assess how well the model is classifying the reviews.
   - A **confusion matrix** is used to analyze the distribution of true positive, true negative, false positive, and false negative predictions.

4. **Performance Testing**:
   - The system is tested on large datasets to ensure that it can handle large volumes of data without significant performance degradation.
   - The time taken for scraping, sentiment analysis, and result generation is measured to ensure the system is efficient.

5. **User Testing**:
   - User feedback is gathered to assess the usability and practicality of the results presentation. This helps refine the user interface and ensures the system provides actionable insights.

# 5.Code and Results

## Source Code:

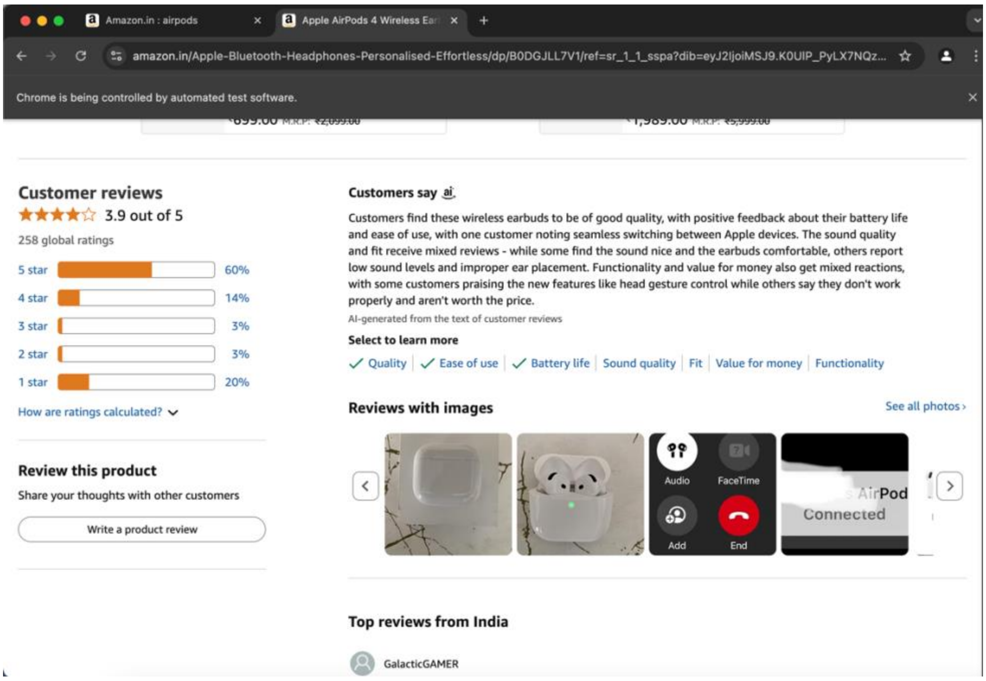
### i.Web Scraping:

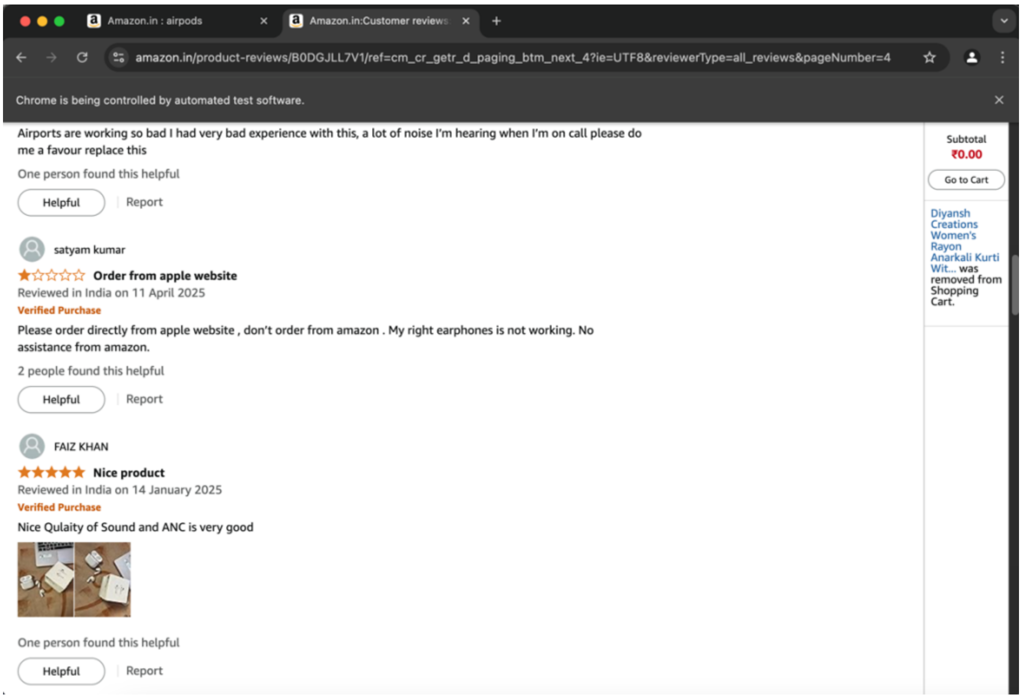
```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
import csv
import re

drive = webdriver.Chrome()
drive.implicitly_wait(5)

drive.get("https://www.amazon.in/")
time.sleep(2)

search = WebDriverWait(drive, 10).until(
   EC.presence_of_element_located((By.XPATH, "//input[@placeholder='Search Amazon.in']"))
)
search.send_keys("airpods")
search.send_keys(Keys.RETURN)
time.sleep(2)

product = WebDriverWait(drive, 10).until(
        EC.element_to_be_clickable((By.XPATH, "//img[@class='s-image']"))
)
product.click()
time.sleep(3)

original_window = drive.current_window_handle
WebDriverWait(drive, 5).until(EC.number_of_windows_to_be(2))
for window_handle in drive.window_handles:
   if window_handle != original_window:
      drive.switch_to.window(window_handle)
      break

try:
   review_button = WebDriverWait(drive, 5).until(
      EC.element_to_be_clickable((By.XPATH, "//a[@id='acrCustomerReviewLink']"))
   )
   review_button.click()
   time.sleep(2)
   print("Navigated to all reviews.")
except:
   print("❌ Review button not found. Proceeding with available reviews...")
```

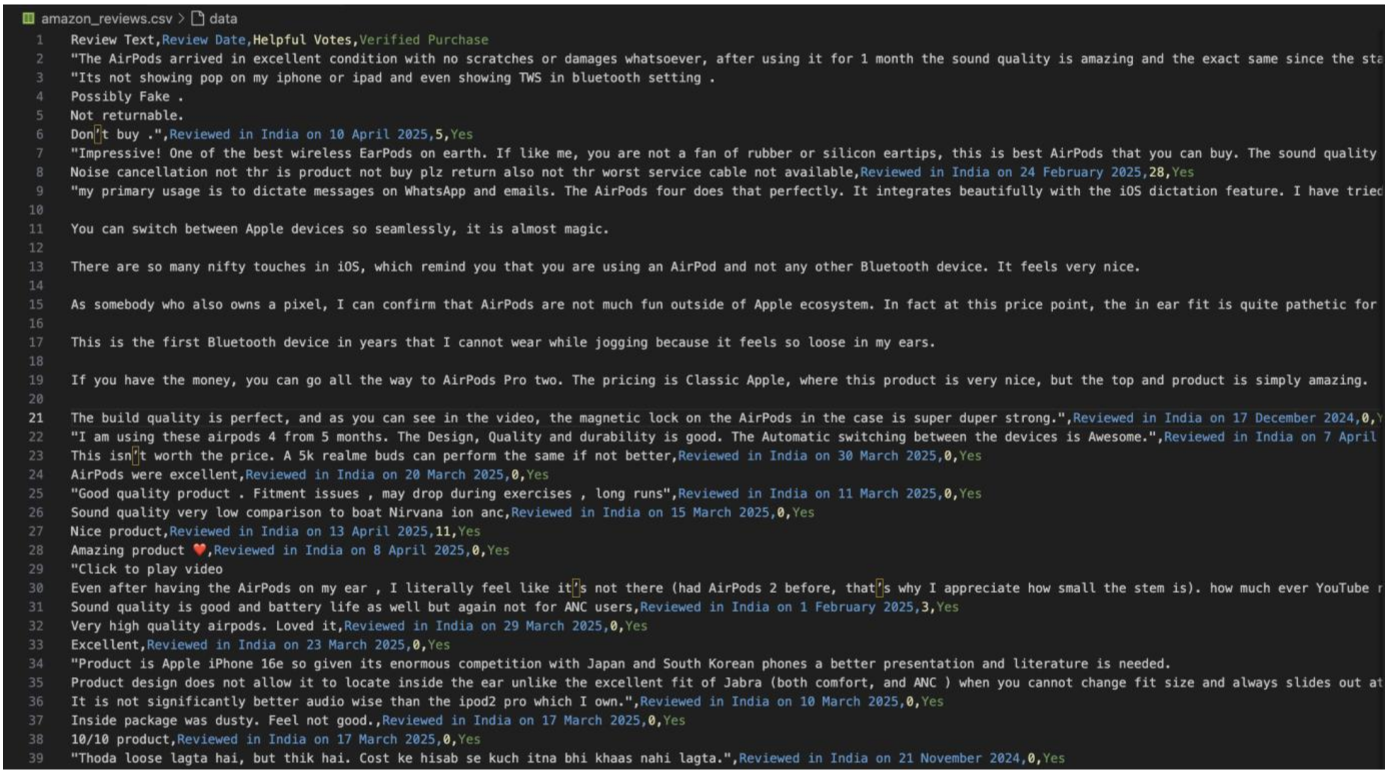
```
#User Login is considered manually
input("Please log in manually, then press Enter to continue...")
print("Continuing with scraping...")

# Open CSV file for writing
with open("amazon_reviews.csv", "w", newline="", encoding="utf-8") as file:
    writer = csv.writer(file)
    writer.writerow(["Review Text", "Review Date", "Helpful Votes", "Verified Purchase"])

    collected_reviews = 0

    while collected_reviews < 1000:
        print(f"Scraping... Total collected: {collected_reviews}")

        # Scroll to load more reviews
        drive.execute_script("window.scrollBy(0, 500);")
        time.sleep(2)

        see_more_buttons = drive.find_elements(By.XPATH, "//span[contains(text(), 'See more')]")
        for btn in see_more_buttons:
            drive.execute_script("arguments[0].click();", btn)
            time.sleep(1)

        reviews = drive.find_elements(By.XPATH, "//span[@data-hook='review-body']")
        dates = drive.find_elements(By.XPATH, "//span[@data-hook='review-date']")
        helpful_votes = drive.find_elements(By.XPATH, "//span[@data-hook='helpful-vote-statement']")
        verified_purchases = drive.find_elements(By.XPATH, "//span[@data-hook='avp-badge']")

        review_texts = [r.text.strip() for r in reviews]
        review_dates = [d.text.strip() for d in dates]
        review_helpful_votes = [re.search(r"(\d+)", h.text).group(1) if re.search(r"(\d+)", h.text) else "0"
for h in helpful_votes]
        verified_purchases_list = ["Yes" if "Verified Purchase" in v.text else "No" for v in
verified_purchases]

        max_length = max(len(review_texts), len(review_dates), len(review_helpful_votes),
len(verified_purchases_list))
        while len(review_texts) < max_length:
            review_texts.append("N/A")
        while len(review_dates) < max_length:
            review_dates.append("N/A")
        while len(review_helpful_votes) < max_length:
            review_helpful_votes.append("0")
        while len(verified_purchases_list) < max_length:
            verified_purchases_list.append("No")

        for text, date, votes, verified in zip(review_texts, review_dates, review_helpful_votes,
verified_purchases_list):
            writer.writerow([text, date, votes, verified])
            collected_reviews += 1
            if collected_reviews >= 1000:
```

```
            break

        try:
            next_button = WebDriverWait(drive, 3).until(
                EC.element_to_be_clickable((By.XPATH, "//li[@class='a-last']/a"))
            )
            drive.execute_script("arguments[0].scrollIntoView();", next_button)
            next_button.click()
            time.sleep(2)
        except:
            print("✅ No more review pages found.")
            break

print("✅ Scraping completed. Data saved in 'amazon_reviews.csv'")

drive.quit()
```

## Outputs:





```
amazon_reviews.csv > data
Review Text,Review Date,Helpful Votes,Verified Purchase
"The AirPods arrived in excellent condition with no scratches or damages whatsoever, after using it for 1 month the sound quality is amazing and the exact same since the sta
"Its not showing pop on my iphone or ipad and even showing TWS in bluetooth setting .
Possibly Fake .
Not returnable.
Don't buy .",Reviewed in India on 10 April 2025,5,Yes
"Impressive! One of the best wireless EarPods on earth. If like me, you are not a fan of rubber or silicon eartips, this is best AirPods that you can buy. The sound quality
Noise cancellation not thr is product not buy plz return also not thr worst service cable not available,Reviewed in India on 24 February 2025,28,Yes
"my primary usage is to dictate messages on WhatsApp and emails. The AirPods four does that perfectly. It integrates beautifully with the iOS dictation feature. I have tried

You can switch between Apple devices so seamlessly, it is almost magic.

There are so many nifty touches in iOS, which remind you that you are using an AirPod and not any other Bluetooth device. It feels very nice.

As somebody who also owns a pixel, I can confirm that AirPods are not much fun outside of Apple ecosystem. In fact at this price point, the in ear fit is quite pathetic for

This is the first Bluetooth device in years that I cannot wear while jogging because it feels so loose in my ears.

If you have the money, you can go all the way to AirPods Pro two. The pricing is Classic Apple, where this product is very nice, but the top and product is simply amazing.

The build quality is perfect, and as you can see in the video, the magnetic lock on the AirPods in the case is super duper strong.",Reviewed in India on 17 December 2024,0,Y
"I am using these airpods 4 from 5 months. The Design, Quality and durability is good. The Automatic switching between the devices is Awesome.",Reviewed in India on 7 April
This isn't worth the price. A 5k realme buds can perform the same if not better,Reviewed in India on 30 March 2025,0,Yes
AirPods were excellent,Reviewed in India on 20 March 2025,0,Yes
"Good quality product . Fitment issues , may drop during exercises , long runs",Reviewed in India on 11 March 2025,0,Yes
Sound quality very low comparison to boat Nirvana ion anc,Reviewed in India on 15 March 2025,0,Yes
Nice product,Reviewed in India on 13 April 2025,11,Yes
Amazing product ❤️,Reviewed in India on 8 April 2025,0,Yes
"Click to play video
Even after having the AirPods on my ear , I literally feel like it's not there (had AirPods 2 before, that's why I appreciate how small the stem is). how much ever YouTube r
Sound quality is good and battery life as well but again not for ANC users,Reviewed in India on 1 February 2025,3,Yes
Very high quality airpods. Loved it,Reviewed in India on 29 March 2025,0,Yes
Excellent,Reviewed in India on 23 March 2025,0,Yes
"Product is Apple iPhone 16e so given its enormous competition with Japan and South Korean phones a better presentation and literature is needed.
Product design does not allow it to locate inside the ear unlike the excellent fit of Jabra (both comfort, and ANC ) when you cannot change fit size and always slides out at
It is not significantly better audio wise than the ipod2 pro which I own.",Reviewed in India on 10 March 2025,0,Yes
Inside package was dusty. Feel not good.,Reviewed in India on 17 March 2025,0,Yes
10/10 product,Reviewed in India on 17 March 2025,0,Yes
"Thoda loose lagta hai, but thik hai. Cost ke hisab se kuch itna bhi khaas nahi lagta.",Reviewed in India on 21 November 2024,0,Yes
```

## ii.Data Cleaning and Sentiment Analysis:

```
import time
import pandas as pd
import spacy
```

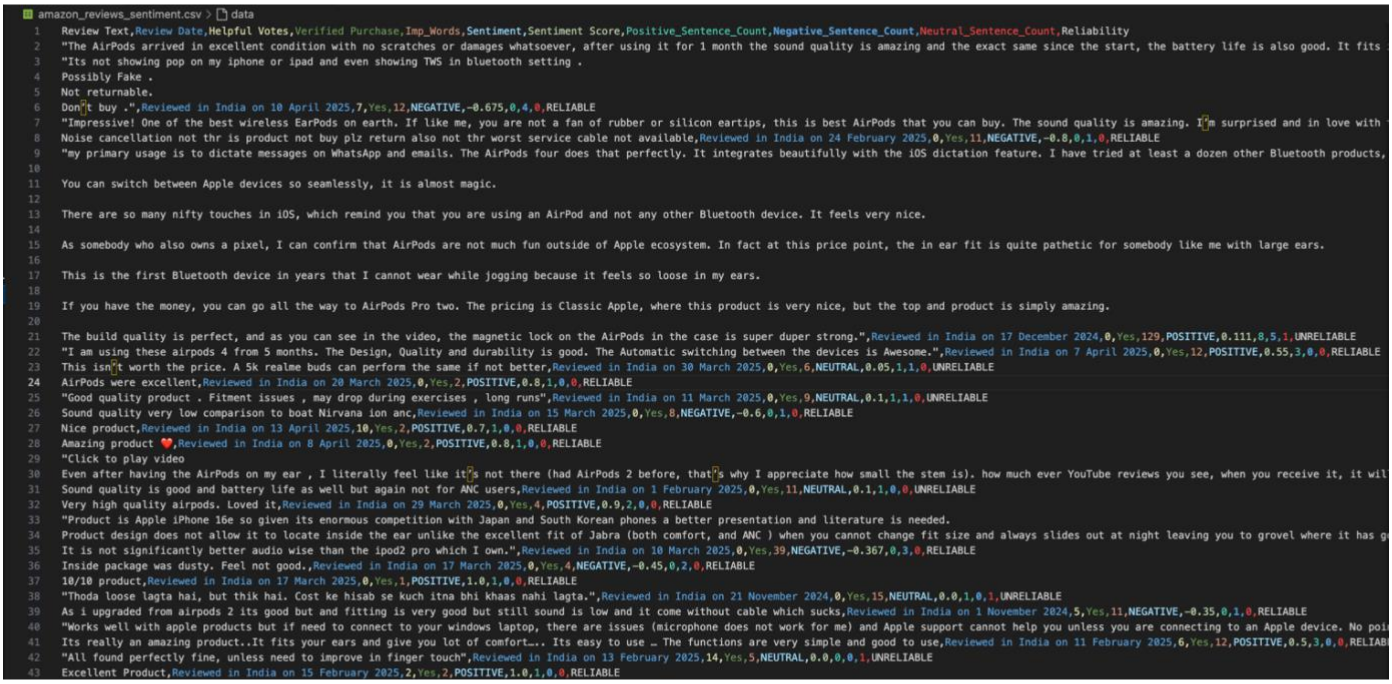
```
import requests

TOGETHER_API_KEY =
"0f204cdb020f5899698bbb8c8d1b12a74dc9ee54103d0272440e182e07037ea3"
TOGETHER_MODEL = "mistralai/Mixtral-8x7B-Instruct-v0.1"
TOGETHER_API_URL = "https://api.together.xyz/v1/chat/completions"

nlp = spacy.load("en_core_web_sm")

def sentence_splitter(text):
    doc = nlp(text)
    return [sent.text.strip() for sent in doc.sents if sent.text.strip()]

def count_non_verbs(text):
    if pd.isnull(text) or not text.strip():
        return 0
    doc = nlp(text)
    excluded_tags = {"VERB", "AUX", "DET", "PRON", "CCONJ", "SCONJ", "PART"}
    imp_words = [token.text for token in doc if token.pos_ not in excluded_tags and token.is_alpha]
    return len(imp_words)

def clean_data(csv_file_path, na_values=None):
    if na_values is None:
        na_values = ["None", "none", "NA", "N/A", "n/a", "null", "NULL", "-", ""]

    print(f"\n📥 Reading file: {csv_file_path}")
    df = pd.read_csv(csv_file_path, na_values=na_values)

    if "Verified Purchase" in df.columns:
        df["Verified Purchase"] = df["Verified Purchase"].apply(
            lambda x: "Yes" if str(x).strip().lower() == "yes" else "No"
        )
        print("🔷 Cleaned 'Verified Purchase' column to Yes/No")

    df.dropna(inplace=True)
    print(f"🔷 Dropped missing values. Shape: {df.shape}")

    if "Review Text" in df.columns:
        original_len = len(df)
        df = df[df["Review Text"].str.len() > 10]
        print(f"🔷 Filtered short reviews (<10 chars): {original_len - len(df)} removed")

    df.drop_duplicates(inplace=True)
    print(f"🔷 Dropped duplicates. Shape: {df.shape}")
    df["Imp_Words"] = df["Review Text"].apply(count_non_verbs)

    return df

def query_together_api(prompt, max_tokens=10):
    headers = {
        "Authorization": f"Bearer {TOGETHER_API_KEY}",
        "Content-Type": "application/json"
    }
```

```
    body = {
        "model": TOGETHER_MODEL,
        "messages": [{"role": "user", "content": prompt}],
        "temperature": 0.3,
        "max_tokens": max_tokens,
        "top_p": 1.0
    }

    try:
        response = requests.post(TOGETHER_API_URL, headers=headers, json=body)
        response.raise_for_status()
        return response.json()["choices"][0]["message"]["content"].strip().upper()
    except Exception as e:
        print(f"❌ API Error: {e}")
        return "ERROR"

def analyze_review_sentiment(review_text, delay=1.2):
    sentences = sentence_splitter(review_text)
    pos_count = 0
    neg_count = 0
    neutral_count = 0
    score_total = 0
    scored_sentences = 0

    for sent in sentences:
        if not sent.strip():
            continue
        prompt = f"""Classify the sentiment of the following sentence. Respond with POSITIVE,
NEGATIVE, or NEUTRAL, followed by a number between -1 and 1 as the sentiment score.

Sentence: "{sent}"
Response:"""
        response = query_together_api(prompt, max_tokens=20)
        label = "NEUTRAL"
        score = 0.0
        if "POSITIVE" in response:
            label = "POSITIVE"
            pos_count += 1
        elif "NEGATIVE" in response:
            label = "NEGATIVE"
            neg_count += 1
        elif "NEUTRAL" in response:
            label = "NEUTRAL"
            neutral_count += 1

        try:
            parts = response.replace(",", "").split()
            score = float([s for s in parts if s.replace('.', '', 1).replace('-', '', 1).isdigit()][-1])
        except Exception:
            score = 0.0

        score_total += score
        scored_sentences += 1
```

```
        time.sleep(delay)

    avg_score = round(score_total / scored_sentences, 3) if scored_sentences > 0 else 0.0

    if avg_score > 0.1:
        sentiment = "POSITIVE"
    elif avg_score < -0.1:
        sentiment = "NEGATIVE"
    else:
        sentiment = "NEUTRAL"

    return sentiment, avg_score, pos_count, neg_count, neutral_count

def determine_reliability_based_on_sentences(pos_count, neg_count, total_sentences, avg_score):
    if abs(avg_score) <= 0.2:  # Neutral score condition
        return "UNRELIABLE"
    else:
        return "RELIABLE"

def apply_detailed_sentiment(df, review_column="Review Text"):
    print("\n🧠 Running Sentiment Analysis...")
    sentiments = []
    scores = []
    positive_counts = []
    negative_counts = []
    neutral_counts = []
    reliability = []

    for i, text in enumerate(df[review_column]):
        print(f"🔍 Processing {i+1}/{len(df)}")
        sentiment, avg_score, pos_count, neg_count, neutral_count = analyze_review_sentiment(text)
        total_sentences = pos_count + neg_count + neutral_count
        sentence_reliability = determine_reliability_based_on_sentences(
            pos_count, neg_count, total_sentences, avg_score)

        sentiments.append(sentiment)
        scores.append(avg_score)
        positive_counts.append(pos_count)
        negative_counts.append(neg_count)
        neutral_counts.append(neutral_count)
        reliability.append(sentence_reliability)
    df["Sentiment"] = sentiments
    df["Sentiment Score"] = scores
    df["Positive_Sentence_Count"] = positive_counts
    df["Negative_Sentence_Count"] = negative_counts
    df["Neutral_Sentence_Count"] = neutral_counts
    df["Reliability"] = reliability

    print("\n📊 Sentiment Summary:")
    print(df["Sentiment"].value_counts())
    return df

if __name__ == "__main__":
```

```python
    input_csv = "amazon_reviews.csv"
    output_clean_csv = "amazon_reviews_cleaned.csv"
    output_sentiment_csv = "amazon_reviews_sentiment.csv"

    df_clean = clean_data(input_csv)
    df_clean.to_csv(output_clean_csv, index=False)
    print(f"\n✅ Cleaned data saved to {output_clean_csv}")

    df_with_sentiment = apply_detailed_sentiment(df_clean)
    df_with_sentiment.to_csv(output_sentiment_csv, index=False)
    print(f"✅ Full sentiment results saved to {output_sentiment_csv}")
```

## Outputs:



```
amazon_reviews_sentiment.csv > data
Review Text,Review Date,Helpful Votes,Verified Purchase,Imp_Words,Sentiment,Sentiment Score,Positive_Sentence_Count,Negative_Sentence_Count,Neutral_Sentence_Count,Reliability
"The AirPods arrived in excellent condition with no scratches or damages whatsoever, after using it for 1 month the sound quality is amazing and the exact same since the start, the battery life is also good. It fits
"Its not showing pop on my iphone or ipad and even showing TWS in bluetooth setting .
Possibly Fake .
Not returnable.
Don’t buy .",Reviewed in India on 10 April 2025,7,Yes,12,NEGATIVE,-0.675,0,4,0,RELIABLE
"Impressive! One of the best wireless EarPods on earth. If like me, you are not a fan of rubber or silicon eartips, this is best AirPods that you can buy. The sound quality is amazing. I’m surprised and in love with
Noise cancellation not thr is product not buy plz return also not thr worst service cable not available,Reviewed in India on 24 February 2025,0,Yes,11,NEGATIVE,-0.8,0,1,0,RELIABLE
"my primary usage is to dictate messages on WhatsApp and emails. The AirPods four does that perfectly. It integrates beautifully with the iOS dictation feature. I have tried at least a dozen other Bluetooth products,

You can switch between Apple devices so seamlessly, it is almost magic.

There are so many nifty touches in iOS, which remind you that you are using an AirPod and not any other Bluetooth device. It feels very nice.

As somebody who also owns a pixel, I can confirm that AirPods are not much fun outside of Apple ecosystem. In fact at this price point, the in ear fit is quite pathetic for somebody like me with large ears.

This is the first Bluetooth device in years that I cannot wear while jogging because it feels so loose in my ears.

If you have the money, you can go all the way to AirPods Pro two. The pricing is Classic Apple, where this product is very nice, but the top and product is simply amazing.

The build quality is perfect, and as you can see in the video, the magnetic lock on the AirPods in the case is super duper strong.",Reviewed in India on 17 December 2024,0,Yes,129,POSITIVE,0.111,8,5,1,UNRELIABLE
"I am using these airpods 4 from 5 months. The Design, Quality and durability is good. The Automatic switching between the devices is Awesome.",Reviewed in India on 7 April 2025,0,Yes,12,POSITIVE,0.55,3,0,0,RELIABLE
This isn’t worth the price. A 5k realme buds can perform the same if not better,Reviewed in India on 30 March 2025,0,Yes,6,NEUTRAL,0.05,1,1,0,UNRELIABLE
AirPods were excellent,Reviewed in India on 20 March 2025,0,Yes,2,POSITIVE,0.8,1,0,0,RELIABLE
"Good quality product . Fitment issues , may drop during exercises , long runs",Reviewed in India on 11 March 2025,0,Yes,9,NEUTRAL,0.1,1,1,0,UNRELIABLE
Sound quality very low comparison to boat Nirvana ion anc,Reviewed in India on 15 March 2025,0,Yes,8,NEGATIVE,-0.6,0,1,0,RELIABLE
Nice product,Reviewed in India on 13 April 2025,10,Yes,2,POSITIVE,0.7,1,0,0,RELIABLE
Amazing product ❤️,Reviewed in India on 8 April 2025,0,Yes,2,POSITIVE,0.8,1,0,0,RELIABLE
"Click to play video
Even after having the AirPods on my ear , I literally feel like it’s not there (had AirPods 2 before, that’s why I appreciate how small the stem is). how much ever YouTube reviews you see, when you receive it, it wil
Sound quality is good and battery life as well but again not for ANC users,Reviewed in India on 1 February 2025,0,Yes,11,NEUTRAL,0.1,1,0,0,UNRELIABLE
Very high quality airpods. Loved it,Reviewed in India on 29 March 2025,0,Yes,4,POSITIVE,0.9,2,0,0,RELIABLE
"Product is Apple iPhone 16e so given its enormous competition with Japan and South Korean phones a better presentation and literature is needed.
Product design does not allow it to locate inside the ear unlike the excellent fit of Jabra (both comfort, and ANC ) when you cannot change fit size and always slides out at night leaving you to grovel where it has g
It is not significantly better audio wise than the ipod2 pro which I own.",Reviewed in India on 10 March 2025,0,Yes,39,NEGATIVE,-0.367,0,3,0,RELIABLE
Inside package was dusty. Feel not good.,Reviewed in India on 17 March 2025,0,Yes,4,NEGATIVE,-0.45,0,2,0,RELIABLE
10/10 product,Reviewed in India on 17 March 2025,0,Yes,1,POSITIVE,1.0,1,0,0,RELIABLE
"Thoda loose lagta hai, but thik hai. Cost ke hisab se kuch itna bhi khaas nahi lagta.",Reviewed in India on 21 November 2024,0,Yes,15,NEUTRAL,0.0,1,0,1,UNRELIABLE
As i upgraded from airpods 2 its good but and fitting is very good but still sound is low and it come without cable which sucks,Reviewed in India on 1 November 2024,5,Yes,11,NEGATIVE,-0.35,0,1,0,RELIABLE
"Works well with apple products but if need to connect to your windows laptop, there are issues (microphone does not work for me) and Apple support cannot help you unless you are connecting to an Apple device. No poi
Its really an amazing product..It fits your ears and give you lot of comfort…. Its easy to use … The functions are very simple and good to use,Reviewed in India on 11 February 2025,6,Yes,12,POSITIVE,0.5,3,0,0,RELIAB
"All found perfectly fine, unless need to improve in finger touch",Reviewed in India on 13 February 2025,14,Yes,5,NEUTRAL,0.0,0,0,1,UNRELIABLE
Excellent Product,Reviewed in India on 15 February 2025,2,Yes,2,POSITIVE,1.0,1,0,0,RELIABLE
```

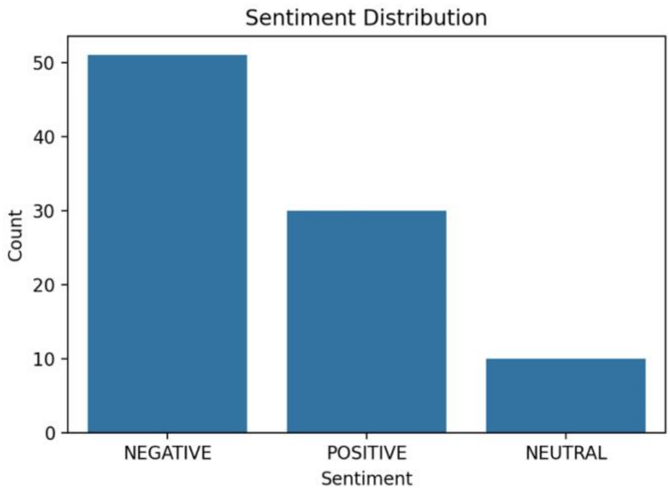
### iii.EDA(Exploratory Data Analysis):

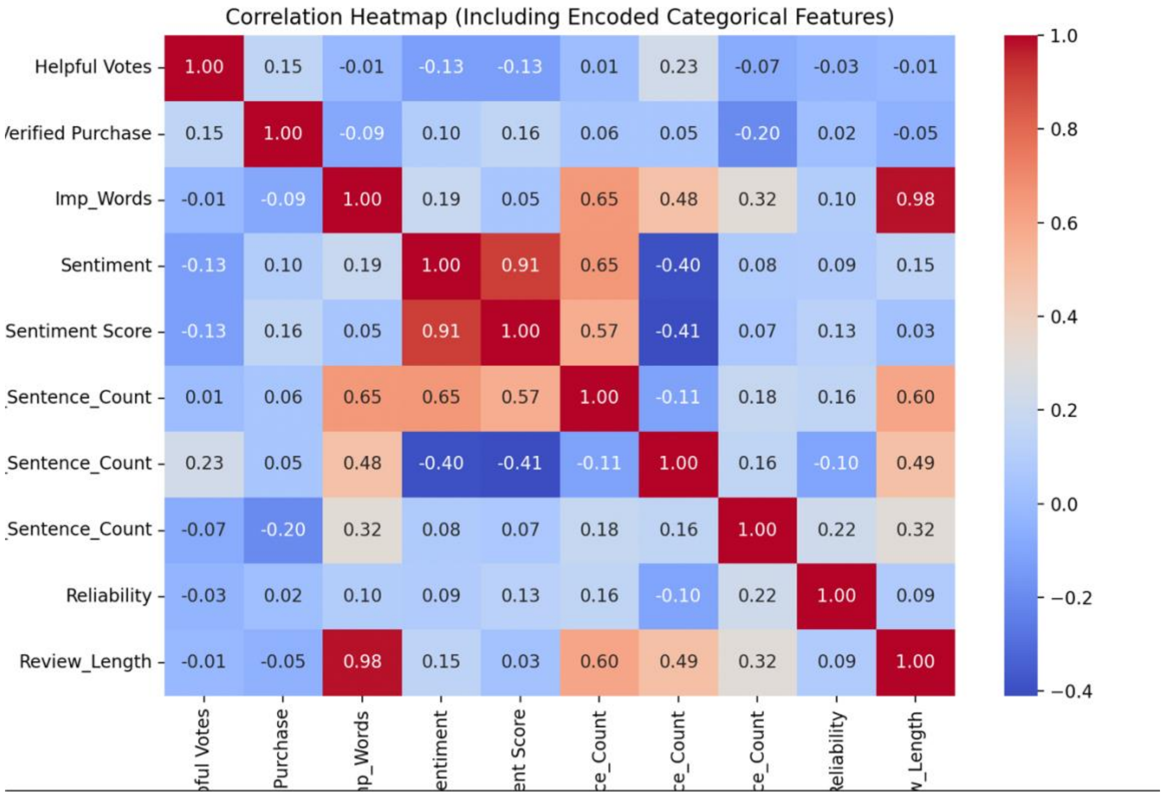
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder

def eda_amazon_reviews(csv_file_path, na_values=None):
    if na_values is None:
        na_values = ["None", "none", "NA", "N/A", "n/a", "null", "NULL", "-", ""]

    print(f"\n📥 Reading file: {csv_file_path}")
    df = pd.read_csv(csv_file_path, na_values=na_values)

    # 1. Basic Info
    print("\n🔷 Basic Info:")
    print(df.info())
    print("\n🔷 First 5 Rows:")
```

```
print(df.head())

# 2. Summary Stats
print("\n🔷 Numeric Summary:")
print(df.describe())
print("\n🔷 Categorical Summary:")
print(df.describe(include='object'))

# 3. Missing Values
print("\n🔷 Missing Values:")
print(df.isnull().sum())

df.dropna(inplace=True)
print(f"\n✅ Dropped NA values. New shape: {df.shape}")

# 4. Duplicates
print(f"\n🔷 Duplicate Rows: {df.duplicated().sum()}")
df.drop_duplicates(inplace=True)
print(f"✅ Dropped duplicates. New shape: {df.shape}")

# 5. Unique values
print("\n🔷 Unique Values Per Column:")
for col in df.columns:
    print(f"{col}: {df[col].nunique()}")

# 6. Review Text Feature
if "Review Text" in df.columns:
    df["Review_Length"] = df["Review Text"].apply(lambda x: len(str(x).split()))
    plt.figure(figsize=(10, 4))
    sns.histplot(df["Review_Length"], bins=50, kde=True)
    plt.title("Distribution of Review Length")
    plt.xlabel("Word Count")
    plt.ylabel("Frequency")
    plt.show()

# 7. Sentiment Distribution
if "Sentiment" in df.columns:
    plt.figure(figsize=(6, 4))
    sns.countplot(data=df, x="Sentiment", order=df["Sentiment"].value_counts().index)
    plt.title("Sentiment Distribution")
    plt.ylabel("Count")
    plt.show()

# 8. Rating Distribution
if "Rating" in df.columns:
    plt.figure(figsize=(6, 4))
    sns.countplot(data=df, x="Rating")
    plt.title("Rating Distribution")
    plt.show()

# 9. Verified Purchase
if "Verified Purchase" in df.columns:
    plt.figure(figsize=(5, 3))
```

```
        sns.countplot(data=df, x="Verified Purchase")
        plt.title("Verified Purchase Distribution")
        plt.show()

    # 10. Correlation Matrix (with encoded categoricals)
    df_encoded = df.copy()

    for col in df_encoded.select_dtypes(include=['object', 'string']):
        if df_encoded[col].nunique() < 20:
            le = LabelEncoder()
            df_encoded[col] = le.fit_transform(df_encoded[col].astype(str))
        else:
            df_encoded.drop(columns=[col], inplace=True)

    corr = df_encoded.corr(numeric_only=True)
    print("\n🔷 Correlation Matrix:")
    print(corr)
    if not corr.empty:
        plt.figure(figsize=(12, 6))
        sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f")
        plt.title("Correlation Heatmap (Including Encoded Categorical Features)")
        plt.show()

    print("\n✅ EDA complete.")
    return df

df = eda_amazon_reviews("amazon_reviews_sentiment.csv")
df.to_csv("cleaned_for_modeling.csv", index=False)
```

## Outputs:






**iv. Model Training:**

```
import pandas as pd
import time
import joblib
from sklearn.model_selection import train_test_split
from sklearn.metrics import (
    accuracy_score, precision_score, recall_score, f1_score
)
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from xgboost import XGBClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.feature_extraction.text import TfidfVectorizer

import warnings
warnings.filterwarnings("ignore")

df = pd.read_csv("cleaned_for_modeling.csv")

text_column = 'Review Text'
label_column = 'Reliability'

label_mapping = {'UNRELIABLE': 0, 'RELIABLE': 1}
df = df[df[label_column].isin(label_mapping)]
y = df[label_column].map(label_mapping)
X_text = df[text_column]
```

```
vectorizer = TfidfVectorizer(stop_words='english', max_features=5000)
X = vectorizer.fit_transform(X_text)

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

models = {
    "Logistic Regression": LogisticRegression(max_iter=1000),
    "Decision Tree": DecisionTreeClassifier(random_state=42),
    "Random Forest": RandomForestClassifier(random_state=42),
    "Support Vector Classifier": SVC(),
    "XGBoost Classifier": XGBClassifier(use_label_encoder=False, eval_metric='mlogloss'),
    "Naive Bayes": MultinomialNB(),
    "KNN Classifier": KNeighborsClassifier()
}

results = []

for name, model in models.items():
    start = time.time()
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    end = time.time()

    results.append({
        "Model": name,
        "Accuracy": round(accuracy_score(y_test, y_pred), 4),
        "Precision": round(precision_score(y_test, y_pred, average='weighted'), 4),
        "Recall": round(recall_score(y_test, y_pred, average='weighted'), 4),
        "F1 Score": round(f1_score(y_test, y_pred, average='weighted'), 4),
        "Train Time (s)": round(end - start, 3)
    })

results_df = pd.DataFrame(results).sort_values(by="F1 Score", ascending=False)

print("\n📊 Classification Model Evaluation Summary:")
print(results_df.to_string(index=False))

#Save the model that is more accurate and has high F1 score along with Less time.
'''knn_model = KNeighborsClassifier()
knn_model.fit(X_train, y_train)
joblib.dump(knn_model, 'knn_model.pkl')
print("KNN model saved successfully")
joblib.dump(vectorizer, 'tfidf_vectorizer.pkl')
print("TF-IDF saved successfully")'''
```

**Outputs:**

```
📊 Classification Model Evaluation Summary:
                     Model  Accuracy  Precision  Recall  F1 Score  Train Time (s)
       Logistic Regression    0.7895     0.6233  0.7895    0.6966           0.005
             Decision Tree    0.7895     0.6233  0.7895    0.6966           0.002
             Random Forest    0.7895     0.6233  0.7895    0.6966           0.045
 Support Vector Classifier    0.7895     0.6233  0.7895    0.6966           0.002
               Naive Bayes    0.7895     0.6233  0.7895    0.6966           0.000
            KNN Classifier    0.7895     0.6233  0.7895    0.6966           0.001
        XGBoost Classifier    0.7368     0.6140  0.7368    0.6699           0.103
```

**v. Sample Prediction:**

```
import joblib
import pandas as pd

model = joblib.load('knn_model.pkl')
vectorizer = joblib.load('tfidf_vectorizer.pkl')

new_reviews = ["I don't like this product. i hate it.","This object is so good."]

X_new = vectorizer.transform(new_reviews)

predicted_reliability = model.predict(X_new)

predictions = []
for reliability in predicted_reliability:
    if reliability == 0:
        predictions.append("Unreliable")
    else:
        predictions.append("Reliable")

for review, reliability in zip(new_reviews, predictions):
    print(f"Review: {review}\nPredicted Reliability: {reliability}\n")
```

## Output:

```
Review: I don't like this product. i hate it.
Predicted Reliability: Reliable

Review: This object is so good.
Predicted Reliability: Reliable
```

# 6. Conclusion & Future Scope

## 6.1 Conclusion:

This project effectively demonstrates an end-to-end pipeline for extracting, analysing, and interpreting customer reviews from Amazon using Selenium and advanced sentiment analysis techniques. By automating the web scraping process and integrating a sentiment classification model, the system delivers actionable insights into customer feedback, including sentiment trends, frequently mentioned keywords, and review behavior patterns.

The incorporation of robust data preprocessing—such as filtering out short and neutral reviews—enhances both the quality and reliability of the results. Overall, this solution offers a scalable and practical tool for businesses seeking to better understand customer sentiment and improve decision-making based on real user feedback.

## 6.2 Future Scope:

i. **Aspect-Based Sentiment Analysis**
Move beyond overall sentiment to analyze sentiment at the aspect level (e.g., price, quality, delivery). This can help pinpoint specific product strengths and weaknesses.

ii. **Fake Review Detection Enhancement**
Integrate advanced techniques (e.g., deep learning, behavior-based models) to improve the detection of fake or biased reviews and ensure more reliable insights.

iii. **Dashboard & Visualization Integration**
Build an interactive dashboard (e.g., using Streamlit or Dash) for real-time visualization of sentiment trends, keyword clouds, and review volumes.

iv. **Voice Review Analysis**
Incorporate audio review sentiment analysis (using speech-to-text + sentiment models) as audio/video reviews become more common.

# 7.References


1. **Selenium WebDriver Documentation:**
   SeleniumHQ. (n.d.). *Selenium with Python Documentation*.
   https://www.selenium.dev/documentation/webdriver/

2. **Scikit-Learn Library :**
   Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). *Scikit-learn: Machine Learning in Python*. *Journal of Machine Learning Research*, 12, 2825-2830.
   https://scikit-learn.org

3. **spaCy NLP Library:**
   ExplosionAI. (n.d.). *spaCy: Industrial-Strength Natural Language Processing in Python*.
   https://spacy.io

4. **TF-IDF Vectorization:**
   Ramos, J. (2003). *Using TF-IDF to Determine Word Relevance in Document Queries*. *Proceedings of the First Instructional Conference on Machine Learning*.

5. **API Key :**
   Together Computer. (n.d.). *Together AI – Open source models and inference APIs*.
   https://www.together.ai

6. **Joblib Documentation:**
   Joblib Developers. (n.d.). *Joblib Documentation*.
   https://joblib.readthedocs.io