“ A Machine learning approach for Ecommerce product evaluation,ranking and recommendation with sentimental analysis”

Sathyadharini S *student, Akshaya S M student, Indra Gandhi Associate Prof,* Department of Information Technology *, College of Engineering Guindy,ANNA UNIVERSITY,Chennai,India.*

Abstract—

Aims to simplify the process of finding the perfect product on an ecommerce site for users by leveraging a combination of machine learning (ML) algorithms and web scraping techniques. By scraping data from Amazon, including product descriptions, URLs, review counts, overall ratings, and specific rating percentages, the system gathers the necessary information. This data is then processed and analyzed using Python, using clustering algorithms for optimal pricing and sentiment analysis performed using Hugging Face's pretrained NLP model.The primary goal of this automation is to save user’s time and provide accurate product recommendations based on ratings, reviews, and customer needs(specifications). Instead of analysing through thousands of products datas on a ecommerce site, users can rely on this system to rank products based on various filters, including data deduplication, data preprocessing, price optimization through clustering, review emotion classification using Hugging Face's pretrained ML model, and sentiment analysis using Vader scores. The final output includes Vader scores along with the top-ranked product's URL, ensuring a streamlined and efficient product selection process.

I. Introduction

In the digital age, e-commerce has revolutionized the way consumers access and purchase products. The convenience of online shopping, however, comes with its own set of challenges, particularly when it comes to finding the perfect product amidst the overwhelming abundance of options. Recognizing this predicament, our research endeavors to simplify and enhance the process of product discovery on e-commerce platforms by harnessing the power of cutting-edge technologies.

This research project aims to present a novel solution that leverages a fusion of machine learning (ML) algorithms and web scraping techniques to streamline the product selection process for users. By employing data extraction methods from a prominent e-commerce platform like Amazon, we gather crucial information, including product descriptions, URLs, review counts, overall ratings, and specific rating percentages. This data forms the foundation upon which our system operates.

The core of our system lies in its ability to process and analyze this vast amount of e-commerce data using Python. Through the application of advanced clustering algorithms, we optimize product pricing, thereby providing users with cost-effective options that align with their preferences. Additionally, sentiment analysis, performed using Hugging Face's pretrained Natural Language Processing (NLP) model, assists in categorizing and understanding customer emotions and opinions expressed in reviews. This multifaceted approach ensures that our system caters to both the analytical and emotional aspects of the user experience.

The primary objective of this automation is to save users' valuable time and offer accurate product recommendations based on an amalgamation of ratings, reviews, and specific customer needs and specifications. Instead of navigating through the daunting task of sifting through thousands of product listings on an e-commerce site, users can rely on our innovative system to efficiently rank products based on a wide array of filters. These filters encompass data deduplication, meticulous data preprocessing, and the unique feature of price optimization through clustering. Furthermore, our sentiment analysis component, powered by Vader scores, enables users to gauge the emotional resonance of product reviews.

In conclusion, this research paper unveils a comprehensive solution that aims to transform the way users discover products on e-commerce platforms. By amalgamating machine learning techniques, web scraping capabilities, and sentiment analysis, we empower consumers with a sophisticated tool to make informed and satisfying purchase decisions. Our system not only simplifies the process but also ensures that users receive top-notch product recommendations tailored to their individual preferences and needs. This research represents a significant step forward in enhancing the e-commerce shopping experience, offering a streamlined, efficient, and user-centric approach to product discovery.

Furthermore, the integration of state-of-the-art technologies such as Hugging Face's pretrained NLP model for sentiment analysis elevates the accuracy and depth of our product recommendations. By understanding the emotional nuances within customer reviews, our system transcends the realm of conventional product filtering and selection, creating a more personalized and satisfying shopping journey.

In an era where time is of the essence and information overload is a common challenge, our research not only contributes to the field of e-commerce but also addresses broader issues of information management and decision-making in the digital age. This paper will delve into the intricacies of our methodology, the results obtained through rigorous testing, and the implications of our system on user experience and satisfaction.

II. Related Work

A. *Webscraping*

In our data-driven world, information is power. Businesses, researchers, and individuals alike rely on vast amounts of data to make informed decisions, gain insights, and enhance their operations. Web scraping, a technique for extracting data from websites, has emerged as a pivotal tool in this data-centric age. This essay explores the concept of web scraping, its applications, challenges, legal considerations, and its evolving role in shaping the digital landscape.Web scraping, also known as web harvesting or web data extraction, refers to the process of automatically collecting data from websites. It involves sending HTTP requests to a website's server, retrieving the HTML content of web pages, and parsing this content to extract the desired information. Web scraping can be performed manually or through automated scripts and tools, making it a versatile method for gathering data from the internet. Web scraping is a powerful tool that unlocks the wealth of information available on the internet. Its applications span across various industries, from business and research to media and finance. However, it also raises legal and ethical considerations that must be carefully addressed.

We have researched and found using Selenium being the most suitable for the web scrapping from any present website comparing to the other scrappers includes Scrapy, Beautiful Soup,Requests, Puppeteer and even ParseHub.

The choice of tool or library depends on factors like the complexity of the scraping task, your programming language preference, and whether you need to scrape static or dynamic websites.Since we have automated web browser of Chrome and scrapped datas from Amazon which is a dynamic website , we have made the use of Selenium and chrome web driver for web automation for scrapping .

B. CLUSTERING /DIVIDING INTO RANGES

Clustering is a fundamental technique in data analysis and machine learning that involves grouping similar data points or objects together based on certain features or characteristics. The goal of clustering is to discover hidden patterns or structures within a dataset and create natural groupings, which can be valuable for various applications, including data exploration, pattern recognition, recommendation systems, and more.

K-Means: One of the most popular centroid-based clustering algorithms. It assigns data points to the cluster with the nearest centroid, where the number of clusters (k) is pre-defined.

Using K-Means clustering for price grouping ranges offers several benefits, especially in retail and pricing strategy optimization and we have used it grouping out the price range and find the range where the percentage of products are more and considering the review counts also here for the optimization.

Clustering is a versatile technique with numerous applications in data analysis and machine learning .The choice of clustering algorithm depends on the nature of the data and the specific problem. . It often requires experimentation and evaluation to determine the most suitable clustering approach for a given task.

C . Hugging Face and Pretrained Models [for emotion classification]

Hugging Face and its pre-trained models democratize access to cutting-edge NLP capabilities, making them accessible to a broad range of users, from researchers and data scientists to developers and organizations. This accessibility, combined with a vibrant community and extensive resources, makes Hugging Face a valuable platform in the field of Natural Language Processing.

Hugging Face models can be used for various NLP tasks, including text classification, sentiment analysis, named entity recognition, machine translation, text generation, and more. By utilizing pre-trained models, organizations can save significant computational resources and time compared to training models from scratch, which is especially valuable for resource-constrained projects. Hugging Face's library supports popular deep learning frameworks like PyTorch and TensorFlow, providing flexibility for users who have preferences or existing infrastructure.

Hence , we have used hugging face ‘s pretrained model ROBERTA MODEL server as nlp model for emotion classification of texts with greater accuracy which has already fined tuned on various datasets.The model was trained using AutoModelForSequenceClassification.from_pretrained with problem_type="multi_label_classification" for 3 epochs with a learning rate of 2e-5 and weight decay of 0.01 with go_emotions for emotions mapping.

https://huggingface.co/SamLowe/roberta-base-go_emotions

D.Natural Language Processing Toolkit[nltk]

The Natural Language Toolkit (NLTK) is a powerful Python library for Natural Language Processing (NLP). It provides a wide range of tools, resources, and functionalities for working with human language data. NLTK is commonly used in various NLP tasks and research, and its uses include:Text Tokenization: NLTK can break down text into words or sentences, a fundamental step in many NLP tasks.Stopword Removal: NLTK includes a list of common stop words (e.g., "the," "and," "in") that can be removed from text to focus on more meaningful words.Part-of-Speech Tagging: NLTK can tag words in a text with their grammatical parts of speech (e.g., nouns, verbs, adjectives), which is essential for tasks like information extraction and sentiment analysis.Hence we used the nltk libraries in python for removing stop words,filtering in the keywords using pos_tag and computing the vader scores for the reviews for sentimental analysis.

III. PROBLEM FORMULATION

A. System Model

In today's digital era, e-commerce platforms offer a vast array of products to consumers. However, the sheer volume of options can overwhelm users, making it challenging to find the perfect product that aligns with their preferences and needs. This problem necessitates a solution that simplifies the process of product discovery and enhances the shopping experience. The core problem addressed in this research can be formulated as follows:

Problem Statement: How can we streamline and optimize the process of product discovery on e-commerce platforms to save users' time and provide them with accurate product recommendations based on ratings, reviews, and customer specifications?

Figure 1 illustrates how our created System model helps out in the problem statement.

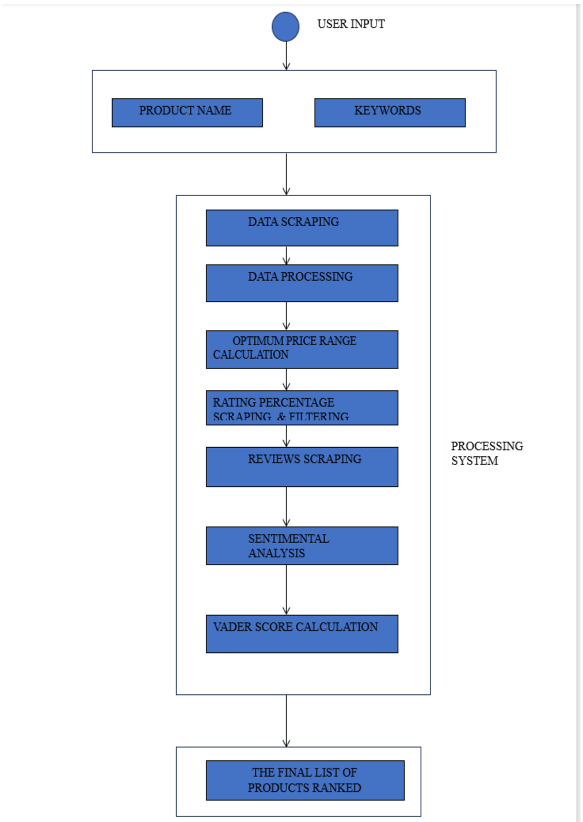


Fig. 1. The Actual System flow/model.

DESCRIBING MODEL FLOW:

Data Acquisition: Gathering comprehensive product data, including product descriptions, URLs, review counts, overall ratings, and specific rating percentages, from e-commerce websites like Amazon.

Data Processing and Analysis: Developing a robust data processing pipeline in Python to preprocess and analyze the collected data efficiently. This includes handling data deduplication and data preprocessing tasks.Machine Learning Algorithms: Implementing machine learning algorithms, particularly clustering algorithms, to optimize product pricing and enhance the product recommendation process.

Sentiment Analysis: Integrating sentiment analysis techniques, including sentiment classification using Hugging Face's pretrained NLP model and Vader scores, to understand and categorize customer emotions and opinions expressed in product reviews.User-Centric Filtering: Creating a user-centric filtering system that allows users to customize product searches based on their specific needs and preferences.

B. Objective

Our objective is to provide an automated solution that simplifies the process of finding the perfect product on e-commerce sites, ultimately enhancing the user's online shopping experience.Develop a web scraping and data collection system to gather essential product information from e-commerce platforms.

Design and implement a data processing pipeline in Python to preprocess and analyze the collected data effectively.Utilize clustering algorithms to optimize product pricing,ensuring cost-effective product recommendations.

Perform sentiment analysis on customer reviews using both Hugging Face's pretrained NLP model and Vader scores to gauge emotional responses.

Create a user-friendly interface that allows users to filter and rank products based on their individual specifications and preferences.

Evaluate the system's performance in terms of accuracy, efficiency, and user satisfaction.

C.Overview of the Proposed Method

In this paper, we propose a novel method to revolutionize the product discovery process on e-commerce platforms, specifically focusing on Amazon, by harnessing a combination of machine learning (ML) algorithms and web scraping techniques. This innovative approach is designed to simplify the user experience, save time, and provide highly accurate product recommendations based on a multitude of factors, including ratings, reviews, and individual customer specifications.

The fig .2. Gives a detailed system architecture and the process taken by our model.

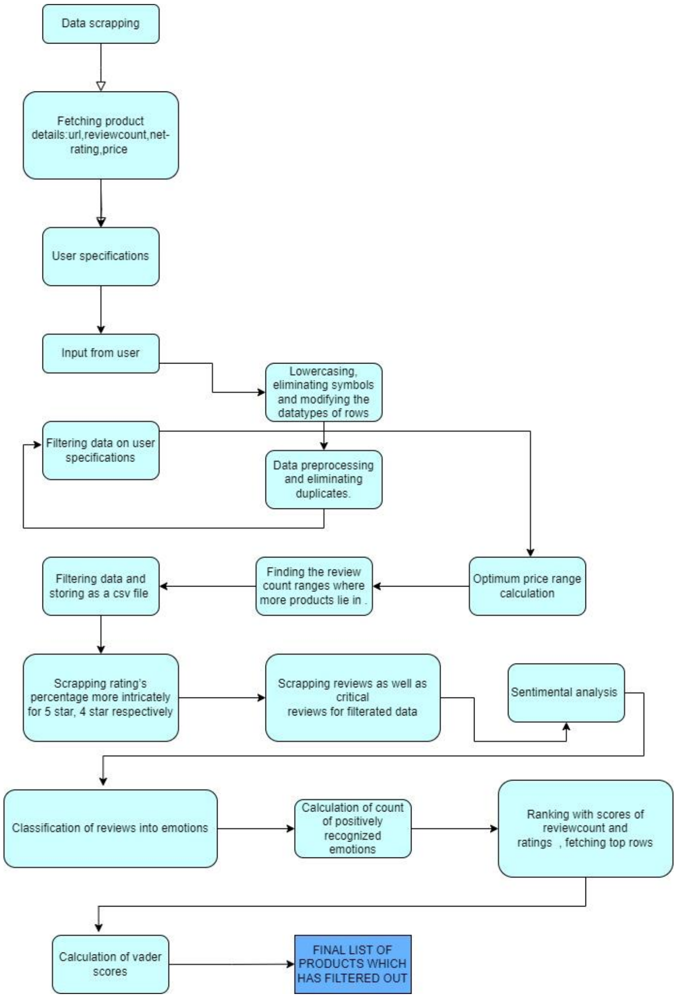

Data scrapping
Fetching product details:url,reviewcount,net-rating,price
User specifications
Input from user
Lowercasing, eliminating symbols and modifying the datatypes of rows
Filtering data on user specifications
Data preprocessing and eliminating duplicates.
Filtering data and storing as a csv file
Finding the review count ranges where more products lie in .
Optimum price range calculation
Scrapping rating's percentage more intricately for 5 star, 4 star respectively
Scrapping reviews as well as critical reviews for filterated data
Sentimental analysis
Classification of reviews into emotions
Calculation of count of positively recognized emotions
Ranking with scores of reviewcount and ratings , fetching top rows
Calculation of vader scores
FINAL LIST OF PRODUCTS WHICH HAS FILTERED OUT

Fig 2. Detailed view of the created model's flow

As a first process,

Data Scraping:This component initiates the process by web scraping Amazon's search results pages using BeautifulSoup and Selenium. It extracts essential product information, including descriptions, prices, ratings, review counts, and URLs, and stores this data in a csv file. The scraper iterates through multiple pages of search results to ensure comprehensive data collection.

User Specifications:After data scraping, the system loads the collected data from the csv file and filters out rows with missing values in critical columns like 'Description', 'Price', 'rating', 'reviewcount', and 'url'. It allows users to input specific keywords to filter products based on their descriptions, providing a user-centric approach to product selection.

Optimum Price Range Calculation (K-Means Clustering):

To optimize pricing, the system calculates the distribution of review counts and divides the data into ranges. It then employs K-means clustering to identify price clusters with the most data points. By considering products with high review counts and their corresponding price ranges, the system filters the data for further analysis.

Scraping Ratings' Percentages and Filtering:

This component scrapes additional data, such as rating percentages for 5-star and 4-star ratings, from Amazon product pages using Selenium. It removes duplicate rows based on the 'Description' column, deduplicates the data, and saves it to a csv file. The scraped data is combined with the original data, including rating percentages, to compute a combined score for each product row. The data is then sorted based on this score and saved to a new csv file.

Reviews Scraping:

The system defines functions to scrape regular and critical reviews from Amazon product pages. It loads product data from the csv file, iterates through the products, and collects their reviews, writing them to separate csv files.

Sentiment Analysis:

Utilizing the Hugging Face Transformers library, sentiment analysis is performed on the scraped reviews. A pre-trained model is employed to classify sentiments within the text. The system analyzes and counts the sentiment labels present in the text data from the csv file, providing insights into the distribution of sentiments within the reviews.

Vader Score Calculation:

The system initializes NLTK resources and reads the regular review data. It preprocesses the text, performs sentiment analysis using the VADER sentiment intensity analyzer, and sums the sentiment compound scores for each review. The summed sentiment scores are printed for each row, providing an additional sentiment perspective.

Final Output:

Based on the calculated VADER scores, the system generates the final product recommendations. The product with the highest VADER score is considered the best-recommended product, as it has received positive sentiment feedback from many Amazon users.

In summary, this comprehensive system integrates web scraping, data preprocessing, clustering, sentiment analysis, and user-centric filtering to streamline the product discovery process on Amazon. It empowers users to make informed purchasing decisions by offering tailored and sentiment-aware product recommendations, ultimately enhancing the e-commerce shopping experience.

IV. *SYSTEM REQUIREMENTS FOR THE PROPOSED MODEL*

System Requirements and Libraries Used:

Python (3.x):The code is written in Python, specifically compatible with Python 3.x versions. Python is the primary programming language used for implementing the entire system

.Beautiful Soup (Bs4):Beautiful Soup is a Python library used for HTML parsing. It allows the system to parse and extract data from HTML pages retrieved during web scraping.

Selenium:Selenium is a crucial library used for browser automation. It enables the system to interact with and automate actions in web browsers like Google Chrome. This is essential for web scraping and data collection.

Web Driver (Chrome):A web driver, specifically for Google Chrome, is used as part of Selenium to automate browser interactions. It provides a way to control and navigate the web browser programmatically.

Pandas:Pandas is a popular data manipulation library in Python. It is used for data cleaning, filtering, and manipulation, making it an integral part of data processing in the system.

Regular Expressions (re):Regular expressions are employed for text pattern matching and extraction. They play a role in data preprocessing and filtering.

Matplotlib.pyplot:Matplotlib is a data visualization library for Python. Matplotlib.pyplot is used for creating various plots and charts to visualize data and results.

Scikit-Learn (sklearn.cluster):Scikit-Learn is a machine learning library in Python. In this system, the sklearn.cluster module is used for implementing clustering algorithms, particularly K-means clustering for price optimization.

OS (os):The OS module provides functions for interacting with the operating system. It may be used for tasks such as file management and directory operations.

Transformers:The Transformers library, likely from Hugging Face, is utilized for natural language processing tasks, specifically sentiment analysis. It leverages pre-trained NLP models to classify sentiments within text data.

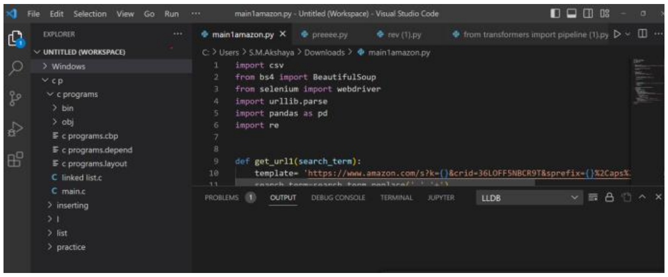
NLTK (nltk):NLTK (Natural Language Toolkit) is a library for working with human language data. In this system, NLTK resources, including a tokenizer, stopwords, and a lemmatizer, are initialized for text preprocessing.

Other Standard Libraries:The system may also utilize other standard libraries and modules as needed for various tasks.

These libraries and system requirements collectively provide the necessary tools and functionality to implement the web scraping, data processing, clustering, sentiment analysis, and visualization components of the system effectively. Each library serves a specific purpose and contributes to the overall functionality and capabilities of the system.

V.OUR IMPLEMENTATION AND RESULTS:

1)importing libraries:



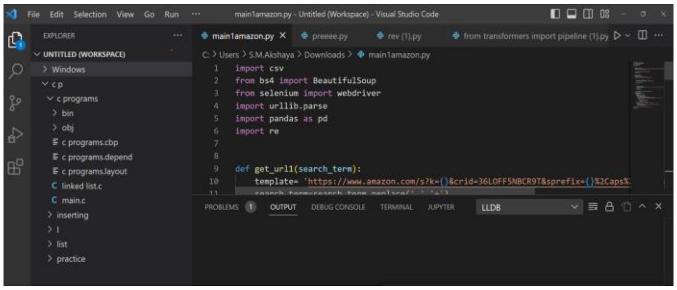
```
import csv
from bs4 import BeautifulSoup
from selenium import webdriver
import urllib.parse
import pandas as pd
import re


def get_url1(search_term):
    template= 'https://www.amazon.com/s?k={}&crid=36LOFF5NBCR9T&sprefix={}%2Caps%
    search_term=search_term.replace(' ','+')
```

This initializes all the necessary libraries to start scrapping the needed datas from the dynamic website,

And we have scrapped from Amazon ecommerce site “www.amazon.com” and produced results since its been widely common.

Importing all the necessary libraries including: csv for saving the data in excel, BeautifulSoup for parsing a web page and Selenium for web automation and a driver has been installed for the Google chrome browser and pandas for data manipulation and analysis and finally importing re ‘regular expressions’ modules.

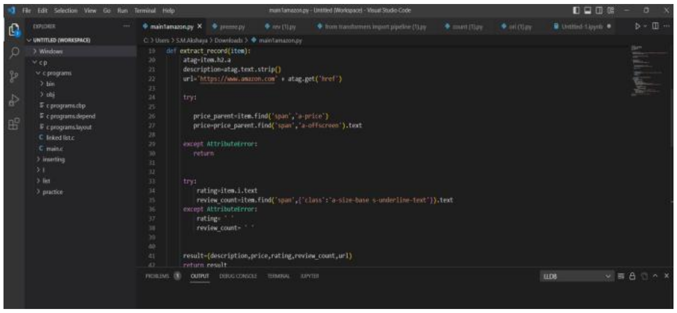
2) get_url1( ) function:



```
import csv
from bs4 import BeautifulSoup
from selenium import webdriver
import urllib.parse
import pandas as pd
import re


def get_url1(search_term):
    template= 'https://www.amazon.com/s?k={}&crid=36LOFF5NBCR9T&sprefix={}%2Caps%
    search_term=search_term.replace(' ','+')
```

Defining a user-defined function named get_url1(search_term) so this helps to process the target url, Where the user specifies the product name, thus we add that as a substring to the common url and hence we get the target url which helps to direct the appropriate site containing the product details.

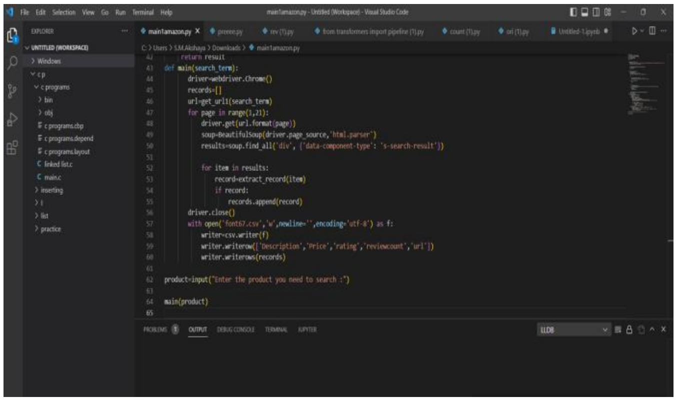
3) “extract_record(item)” function:



```
def extract_record(item):
    atag=item.h2.a
    description=atag.text.strip()
    url='https://www.amazon.com' + atag.get('href')

    try:

      price_parent=item.find('span','a-price')
      price=price_parent.find('span','a-offscreen').text

    except AttributeError:
      return


    try:
       rating=item.i.text
       review_count=item.find('span',{'class':'a-size-base s-underline-text'}).text
    except AttributeError:
       rating= ' '
       review_count= ' '


    result=(description,price,rating,review_count,url)
    return result
```

This function used to scrap the datas :
”description”, “price”,”url”,“review count” and the overall “rating” of the product.

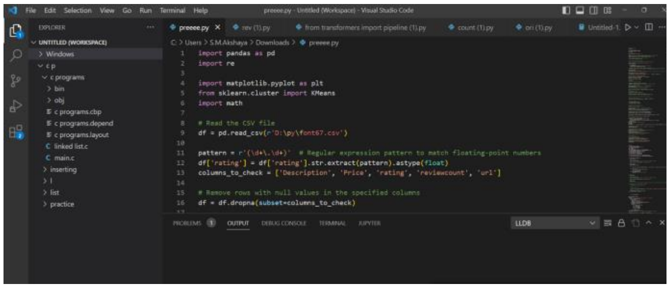
4) main(search_term) function:



```
    return result
def main(search_term):
    driver=webdriver.Chrome()
    records=[]
    url=get_url1(search_term)
    for page in range(1,21):
        driver.get(url.format(page))
        soup=BeautifulSoup(driver.page_source,'html.parser')
        results=soup.find_all('div', {'data-component-type': 's-search-result'})

        for item in results:
            record=extract_record(item)
            if record:
                records.append(record)
    driver.close()
    with open('font67.csv','w',newline='',encoding='utf-8') as f:
        writer=csv.writer(f)
        writer.writerow(['Description','Price','rating','reviewcount','url'])
        writer.writerows(records)

product=input("Enter the product you need to search :")

main(product)
```

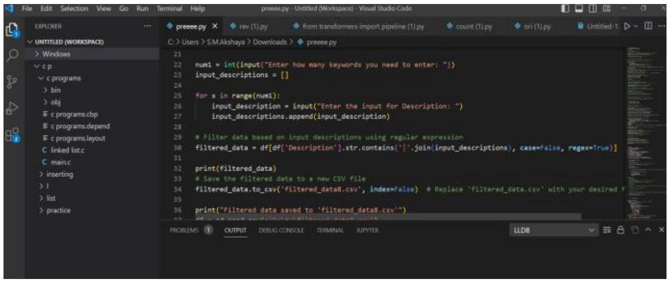
This function main almost scraps a maximum of 21 pages of details containing product details
and storing all the datas in a excel sheet, this traverses to each and every page and scraps one by one.

5)Data Filtering and Price Clustering



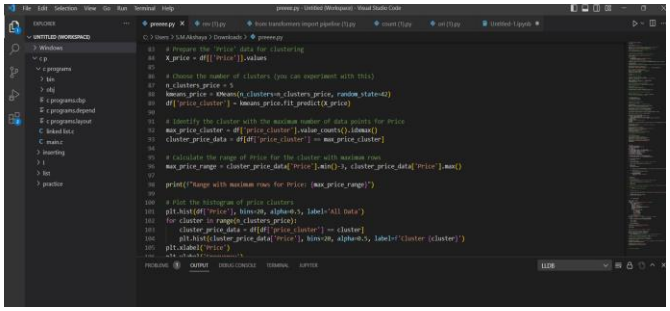
```
import pandas as pd
import re

import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
import math

# Read the CSV file
df = pd.read_csv(r'D:\py\font67.csv')

pattern = r'(\d+\.\d+)'  # Regular expression pattern to match floating-point numbers
df['rating'] = df['rating'].str.extract(pattern).astype(float)
columns_to_check = ['Description', 'Price', 'rating', 'reviewcount', 'url']

# Remove rows with null values in the specified columns
df = df.dropna(subset=columns_to_check)
```

Preprocessing the datas that has been already stored to “font67.csv” and converting them to the appropriate data types that helps in the further processing.



```
num1 = int(input("Enter how many keywords you need to enter: "))
input_descriptions = []

for x in range(num1):
    input_description = input("Enter the input for Description: ")
    input_descriptions.append(input_description)

# Filter data based on input descriptions using regular expression
filtered_data = df[df['Description'].str.contains('|'.join(input_descriptions), case=False, regex=True)]

print(filtered_data)
# Save the filtered data to a new CSV file
filtered_data.to_csv('filtered_data8.csv', index=False)  # Replace 'filtered_data.csv' with your desired f

print("Filtered data saved to 'filtered_data8.csv'")
```

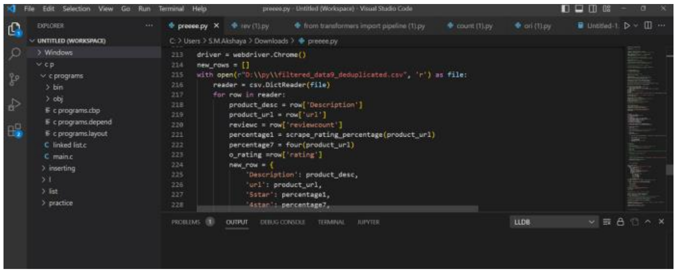
This part asks the user to enter the number of keywords and the keywords also thus using these keywords, we check up in all the product’s description which contain either of these keywords specified and filtering accordingly.



```
# Prepare the 'Price' data for clustering
X_price = df[['Price']].values

# Choose the number of clusters (you can experiment with this)
n_clusters_price = 5
kmeans_price = KMeans(n_clusters=n_clusters_price, random_state=42)
df['price_cluster'] = kmeans_price.fit_predict(X_price)

# Identify the cluster with the maximum number of data points for Price
max_price_cluster = df['price_cluster'].value_counts().idxmax()
cluster_price_data = df[df['price_cluster'] == max_price_cluster]

# Calculate the range of Price for the cluster with maximum rows
max_price_range = cluster_price_data['Price'].min()-3, cluster_price_data['Price'].max()

print(f"Range with maximum rows for Price: {max_price_range}")

# Plot the histogram of price clusters
plt.hist(df['Price'], bins=20, alpha=0.5, label='All Data')
for cluster in range(n_clusters_price):
    cluster_price_data = df[df['price_cluster'] == cluster]
    plt.hist(cluster_price_data['Price'], bins=20, alpha=0.5, label=f'Cluster {cluster}')
plt.xlabel('Price')
```

First grouping them based on the review count and creating 5 ranges of rc’s and counting the number as well as percentages of products falling to each range.Using the Scikit-Learn library's KMeans clustering algorithm to cluster data based on the 'Price' column into a specified number of clusters (n_clusters_price).Identifying the cluster with the maximum number of data points and calculating the price range for the cluster with the maximum rows.

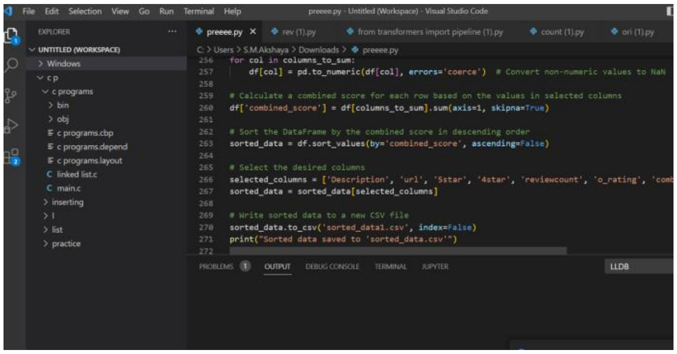
6)Intricate filtering as a step:



```
driver = webdriver.Chrome()
new_rows = []
with open(r"D:\\py\\filtered_data9_deduplicated.csv", 'r') as file:
    reader = csv.DictReader(file)
    for row in reader:
        product_desc = row['Description']
        product_url = row['url']
        reviewc = row['reviewcount']
        percentage1 = scrape_rating_percentage(product_url)
        percentage7 = four(product_url)
        o_rating =row['rating']
        new_row = {
            'Description': product_desc,
            'url': product_url,
            '5star': percentage1,
            '4star': percentage7,
```

Now scrapping the individual rating percentages including 5 star and 4 star and thus this helps in filtering the more product intricately.

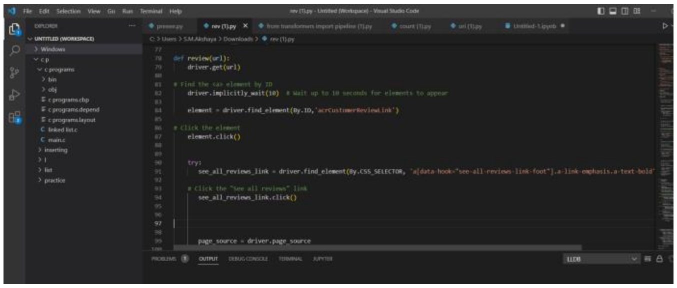
7) Combining and Sorting data:



```
for col in columns_to_sum:
    df[col] = pd.to_numeric(df[col], errors='coerce')  # Convert non-numeric values to NaN

# Calculate a combined score for each row based on the values in selected columns
df['combined_score'] = df[columns_to_sum].sum(axis=1, skipna=True)

# Sort the DataFrame by the combined score in descending order
sorted_data = df.sort_values(by='combined_score', ascending=False)

# Select the desired columns
selected_columns = ['Description', 'url', '5star', '4star', 'reviewcount', 'o_rating', 'comb
sorted_data = sorted_data[selected_columns]

# Write sorted data to a new CSV file
sorted_data.to_csv('sorted_data1.csv', index=False)
print("Sorted data saved to 'sorted_data.csv'")
```

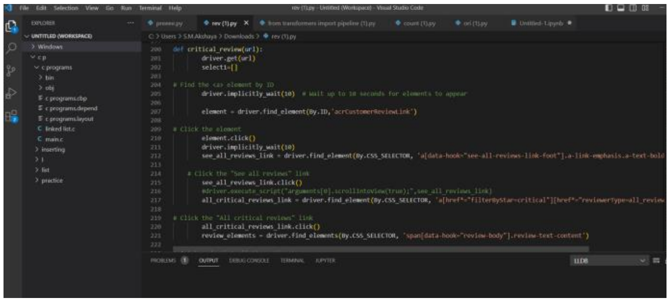
Now finding the combined score of reviewcount , 5 star , 4 star and calculating the overall score(summation of three values) which helps in the ranking of the products hence , this process helps to filter the product and saving into new csv file .

8) Scraping Reviews



```
def review(url):
    driver.get(url)

# Find the <a> element by ID
    driver.implicitly_wait(10)  # Wait up to 10 seconds for elements to appear

    element = driver.find_element(By.ID,'acrCustomerReviewLink')

# Click the element
    element.click()


    try:
        see_all_reviews_link = driver.find_element(By.CSS_SELECTOR, 'a[data-hook="see-all-reviews-link-foot"].a-link-emphasis.a-text-bold'

    # Click the "See all reviews" link
        see_all_reviews_link.click()



        page_source = driver.page_source
```

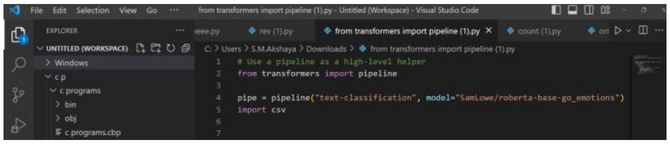
Now scrapping the reviews of the filtered data and this almost 50 -60 reviews of each product and get stored on a csv file.



```
def critical_review(url):
        driver.get(url)
        selecti=[]

# Find the <a> element by ID
        driver.implicitly_wait(10)  # Wait up to 10 seconds for elements to appear

        element = driver.find_element(By.ID,'acrCustomerReviewLink')

# Click the element
        element.click()
        driver.implicitly_wait(10)
        see_all_reviews_link = driver.find_element(By.CSS_SELECTOR, 'a[data-hook="see-all-reviews-link-foot"].a-link-emphasis.a-text-bold

    # Click the "See all reviews" link
        see_all_reviews_link.click()
        #driver.execute_script("arguments[0].scrollIntoView(true);",see_all_reviews_link)
        all_critical_reviews_link = driver.find_element(By.CSS_SELECTOR, 'a[href*="filterByStar=critical"][href*="reviewerType=all_review

# Click the "All critical reviews" link
        all_critical_reviews_link.click()
        review_elements = driver.find_elements(By.CSS_SELECTOR, 'span[data-hook="review-body"].review-text-content')
```

This helps in the scrapping of reviews which are categorized under a tag named “critical reviews”.

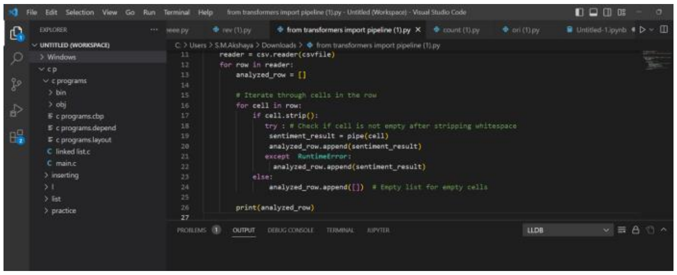
9)Emotion Classification of reviews (Hugging Face’s API)



```
# Use a pipeline as a high-level helper
from transformers import pipeline

pipe = pipeline("text-classification", model="SamLowe/roberta-base-go_emotions")
import csv
```

Pretrained Roberta model has been used which has linked using application interface using hugging face , thus which has a ability to categorize the reviews to several emotion tag namely joy, sad, disappointment and more on..,

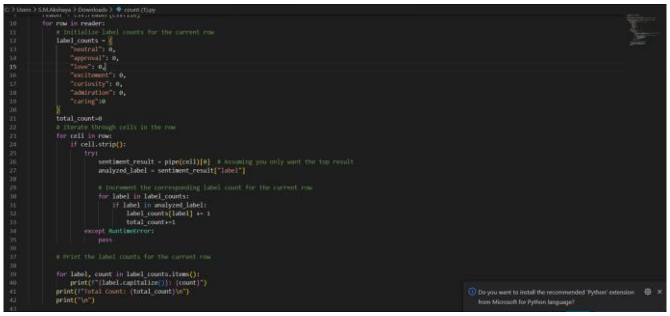
10)Labelling the reviews



```
    reader = csv.reader(csvfile)
    for row in reader:
        analyzed_row = []

        # Iterate through cells in the row
        for cell in row:
            if cell.strip():
                try : # Check if cell is not empty after stripping whitespace
                 sentiment_result = pipe(cell)
                 analyzed_row.append(sentiment_result)
                except  RuntimeError:
                  analyzed_row.append(sentiment_result)
            else:
                analyzed_row.append([])  # Empty list for empty cells

        print(analyzed_row)
```

Each and every review of all recently filtered products will be categorized into a emotion and labelled.

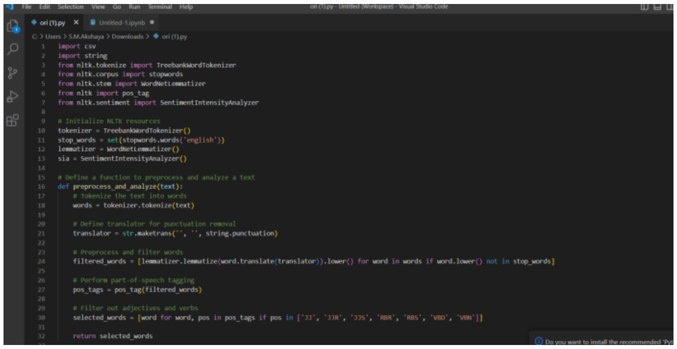
11)Counting and Analysis



```
        for row in reader:
            # Initialize label counts for the current row
            label_counts = {
                "neutral": 0,
                "approval": 0,
                "love": 0,
                "excitement": 0,
                "curiosity": 0,
                "admiration": 0,
                "caring":0
            }
            total_count=0
            # Iterate through cells in the row
            for cell in row:
                if cell.strip():
                    try:
                        sentiment_result = pipe(cell)[0]  # Assuming you only want the top result
                        analyzed_label = sentiment_result["label"]

                        # Increment the corresponding label count for the current row
                        for label in label_counts:
                            if label in analyzed_label:
                                label_counts[label] += 1
                                total_count+=1
                    except RuntimeError:
                        pass

            # Print the label counts for the current row

            for label, count in label_counts.items():
                print(f"{label.capitalize()}: {count}")
            print(f"Total Count: {total_count}\n")
            print("\n")
```

Counting the number of happy emotions that got labelled for the scrapped reviews for every product(recently filtered).

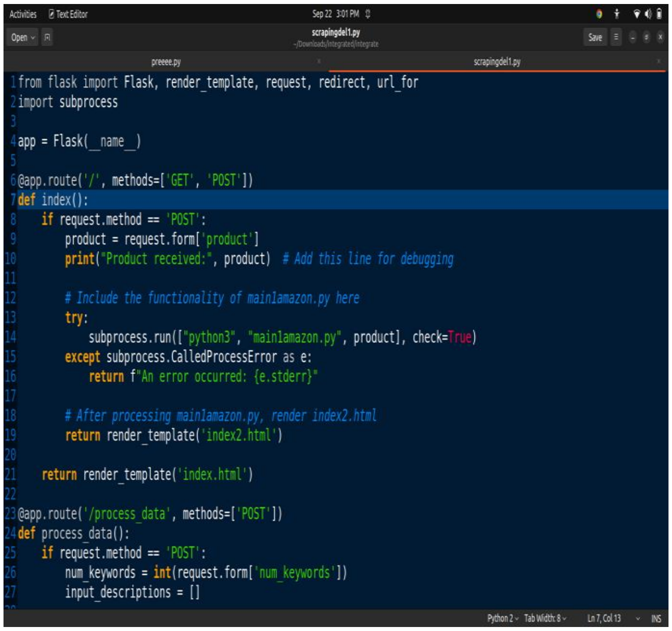
12)NLTK LIBRARY UTILIZATION



```
import csv
import string
from nltk.tokenize import TreebankWordTokenizer
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk import pos_tag
from nltk.sentiment import SentimentIntensityAnalyzer

# Initialize NLTK resources
tokenizer = TreebankWordTokenizer()
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()
sia = SentimentIntensityAnalyzer()

# Define a function to preprocess and analyze a text
def preprocess_and_analyze(text):
    # Tokenize the text into words
    words = tokenizer.tokenize(text)

    # Define translator for punctuation removal
    translator = str.maketrans('', '', string.punctuation)

    # Preprocess and filter words
    filtered_words = [lemmatizer.lemmatize(word.translate(translator)).lower() for word in words if word.lower() not in stop_words]

    # Perform part-of-speech tagging
    pos_tags = pos_tag(filtered_words)

    # Filter out adjectives and verbs
    selected_words = [word for word, pos in pos_tags if pos in ['JJ', 'JJR', 'JJS', 'RBR', 'RBS', 'VBD', 'VBN']]

    return selected_words
```

The code defines a function preprocess_and_analyze to tokenize, preprocess, and analyze text data from a CSV file. It removes punctuation, filters out stop words, and performs part-of-speech tagging to select adjectives, adverbs, and verbs. For each cell in the CSV file, it preprocesses the text, calculates sentiment scores using VADER, and sums the scores for the entire row, then prints the sum of Vader scores for each row.

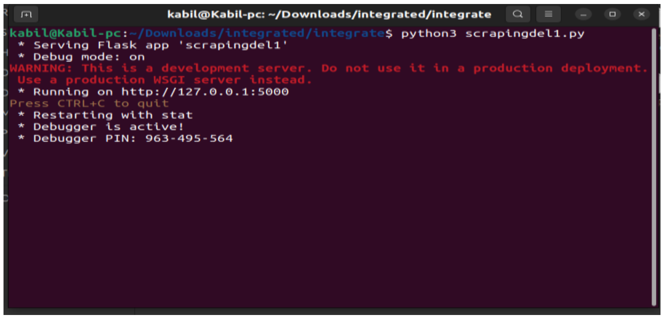
13)Deployment using Flask



```
from flask import Flask, render_template, request, redirect, url_for
import subprocess

app = Flask(__name__)

@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        product = request.form['product']
        print("Product received:", product)  # Add this line for debugging

        # Include the functionality of mainlamazon.py here
        try:
            subprocess.run(["python3", "mainlamazon.py", product], check=True)
        except subprocess.CalledProcessError as e:
            return f"An error occurred: {e.stderr}"

        # After processing mainlamazon.py, render index2.html
        return render_template('index2.html')

    return render_template('index.html')

@app.route('/process_data', methods=['POST'])
def process_data():
    if request.method == 'POST':
        num_keywords = int(request.form['num_keywords'])
        input_descriptions = []
```

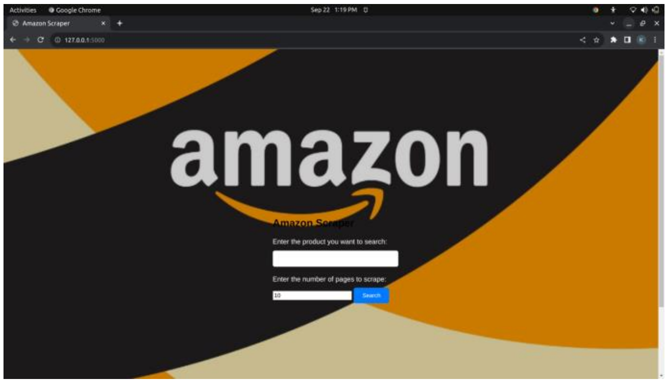
Flask is a lightweight and flexible web framework for building web applications in Python. It is often referred to as a microframework because it provides the essential tools and components for creating web applications and we have made use of this to deploy our model as final step by which the user can able to interact with the created model.

VI.RESULTS /OUTPUTS:



```
kabil@Kabil-pc: ~/Downloads/integrated/integrate
kabil@Kabil-pc:~/Downloads/integrated/integrate$ python3 scrapingdel1.py
 * Serving Flask app 'scrapingdel1'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment.
 Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 963-495-564
```

The model has been compiled and tested in the linux ‘s ubuntu terminal operating system and all installations were made according to the os.



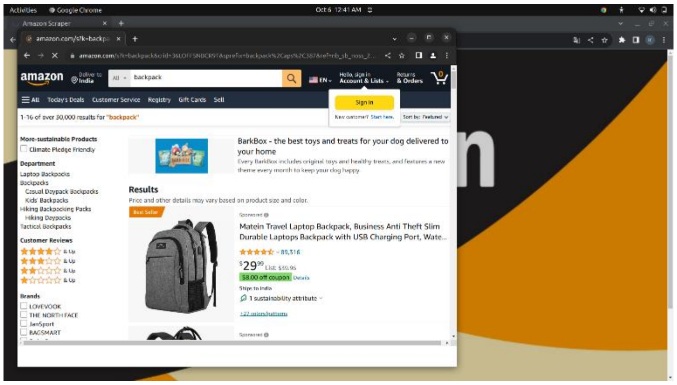
Amazon Scraper

Enter the product you want to search:

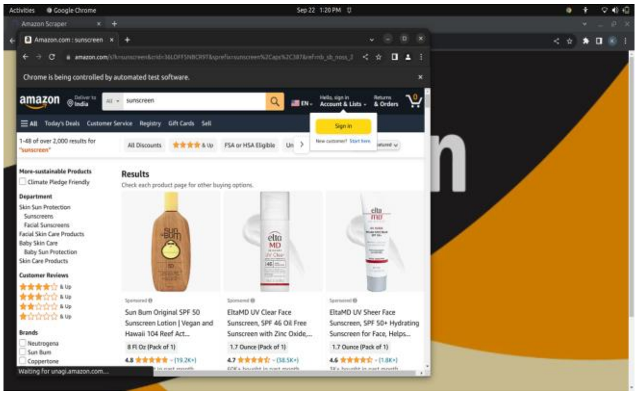
Enter the number of pages to scrape: 10

The front-end that we have incorporated with the flask which deploys where the user gives the input of the product name which is fed in input of amazon’s search box.

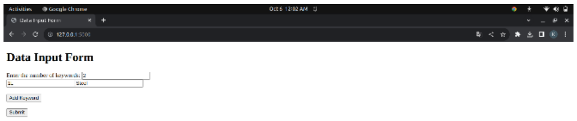
LIVE-PRESENT DATA SCRAPPING:


Activities
Google Chrome
Oct 6 12:41 AM
Amazon Scraper
amazon.com/s?k=backpack
amazon
backpack
Sign in
Results
Matein Travel Laptop Backpack, Business Anti Theft Slim Durable Laptops Backpack with USB Charging Port, Wate...
$29.99
BarkBox - the best toys and treats for your dog delivered to your home
1-16 of over 30,000 results for "backpack"

When backpack as the product name


Activities
Google Chrome
Sep 22 1:20 PM
Amazon Scraper
Amazon.com : sunscreen
Chrome is being controlled by automated test software.
amazon
sunscreen
Sign in
1-48 of over 2,000 results for "sunscreen"
Results
Check each product page for other buying options.
Sun Bum Original SPF 50 Sunscreen Lotion | Vegan and Hawaii 104 Reef Act...
EltaMD UV Clear Face Sunscreen, SPF 46 Oil Free Sunscreen with Zinc Oxide,...
EltaMD UV Sheer Face Sunscreen, SPF 50+ Hydrating Sunscreen for Face, Helps...
Waiting for unagi.amazon.com...

Scrapping datas when Sunscreen as the product name entered by the user

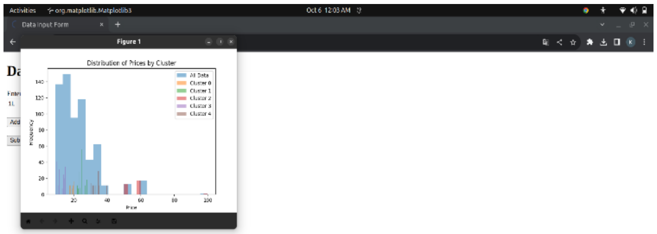


Data Input Form

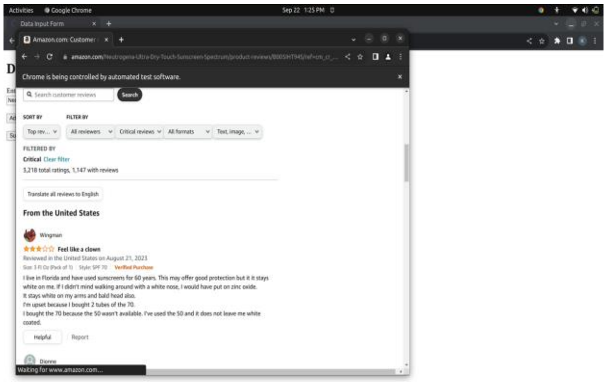
Enter the number of keywords: 2

1L Steel

When Specification entered by the user when water bottle as the product’s name.



This picture shows the distributions of prices visually in graphs and which are classified by k-means clustering algorithm considering products with high review counts and grouping them into ranges and considering the price ranges where most of the products lie in ,and filtering the datas in a separate csv and proceeding with the next process for filtering out.



Chrome is being controlled by automated test software.

SORT BY: Top rev... | FILTER BY: All reviewers, Critical reviews, All formats, Text, image, ...

FILTERED BY

Critical Clear filter

3,218 total ratings, 1,147 with reviews

Translate all reviews to English

From the United States

Wingman

★★★☆☆ Feel like a clown

Reviewed in the United States on August 21, 2023

Size: 3 Fl Oz (Pack of 1) | Style: SPF 70 | Verified Purchase

I live in Florida and have used sunscreens for 60 years. This may offer good protection but it it stays white on me. If I didn't mind walking around with a white nose, I would have put on zinc oxide.
It stays white on my arms and bald head also.
I'm upset because I bought 2 tubes of the 70.
I bought the 70 because the 50 wasn't available. I've used the 50 and it does not leave me white coated.

Helpful | Report

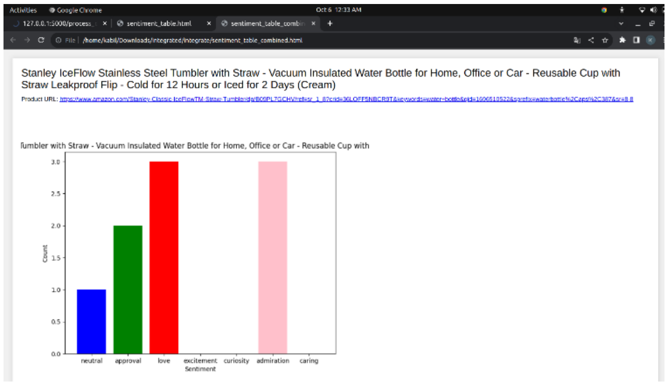
Dionne

Waiting for www.amazon.com...

Scrapping reviews when Sunscreen as the product name entered by the user

Scrapping all the reviews , even the critical reviews which are already categorized by Amazon and saving them in a csv , we scrap almost 40-50 reviews for each product and classifying them into emotion using the Roberta pretrained model which classifies them into emotions : admiration, amusement, anger,annoyance,approval, caring, desire, confusion, curiosity,disgust, excitement, disappointment,joy, love and more on.

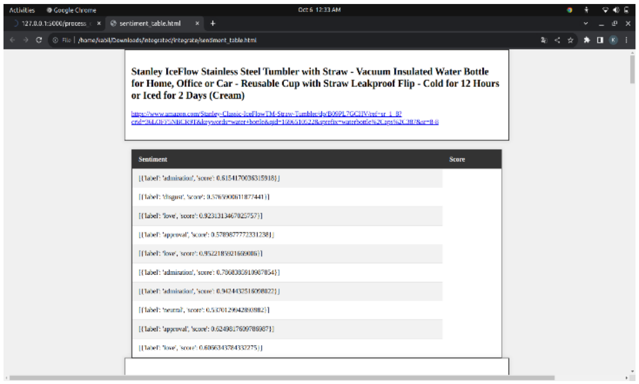
And counting the happy emotions alone i.e., neutral,approval,love,excitement,curiosity, admiration,caring for each product and classifying them which gets the higher score.And finally taking the reviews of those filtered and removing all the stop words and using lemmatizers and tokenizers from the "Natural Language Tool kit" and even performing part of speech tagging with the reviews i.e,.taking /considering all the adjectives and verbs and calculating the vader sum and hence by comparing the resultant vader sum with all the products , we get the best recommended product that many people were happy about it and considering using it.



Stanley IceFlow Stainless Steel Tumbler with Straw - Vacuum Insulated Water Bottle for Home, Office or Car - Reusable Cup with Straw Leakproof Flip - Cold for 12 Hours or Iced for 2 Days (Cream)

Product URL: https://www.amazon.com/Stanley-Classic-IceFlowTM-Straw-Tumbler/dp/B09PL7GCHV/ref=sr_1_8?crid=36LOFF5NBCR9T&keywords=water+bottle&qid=1696510522&sprefix=waterbottle%2Caps%2C387&sr=8-8

Creating plots to visualize the distribution of reviews categorized into different emotional labels.



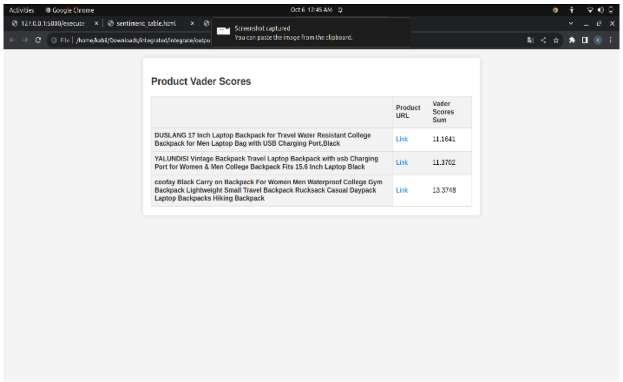
Stanley IceFlow Stainless Steel Tumbler with Straw - Vacuum Insulated Water Bottle for Home, Office or Car - Reusable Cup with Straw Leakproof Flip - Cold for 12 Hours or Iced for 2 Days (Cream)

https://www.amazon.com/Stanley-Classic-IceFlowTM-Straw-Tumbler/dp/B09PL7GCHV/ref=sr_1_8?crid=36LOFF5NBCR9T&keywords=water+bottle&qid=1696510522&sprefix=waterbottle%2Caps%2C387&sr=8-8

Sentiment	Score
[{'label': 'admiration', 'score': 0.6154170036315918}]	
[{'label': 'disgust', 'score': 0.5765900611877441}]	
[{'label': 'love', 'score': 0.9231313467025757}]	
[{'label': 'approval', 'score': 0.5789877772331238}]	
[{'label': 'love', 'score': 0.9522185921669006}]	
[{'label': 'admiration', 'score': 0.7868385910987854}]	
[{'label': 'admiration', 'score': 0.9424432516098022}]	
[{'label': 'neutral', 'score': 0.5370129942893982}]	
[{'label': 'approval', 'score': 0.6249817609786987}]	
[{'label': 'love', 'score': 0.6056343784332275}]	

Emotion label with scores using the Roberta pretrained model.

RESULTS WITH THE EXACT URLS OF THE PRODUCT WITH THE NET VADER SUMS OF REVIEWS:



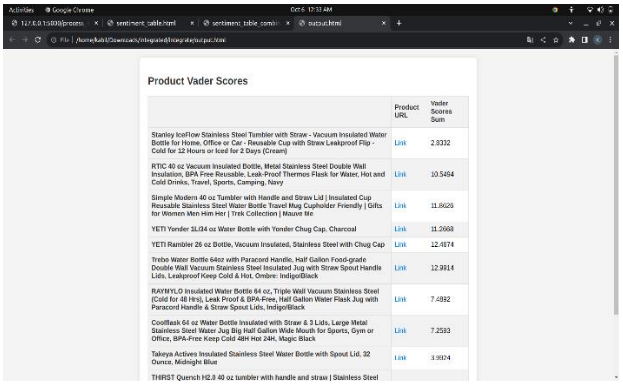
Screenshot captured
You can paste the image from the clipboard.

Product Vader Scores

	Product URL	Vader Scores Sum
DUSLANG 17 Inch Laptop Backpack for Travel Water Resistant College Backpack for Men Laptop Bag with USB Charging Port,Black	Link	11.1641
YALUNDISI Vintage Backpack Travel Laptop Backpack with usb Charging Port for Women & Men College Backpack Fits 15.6 Inch Laptop Black	Link	11.3702
coofay Black Carry on Backpack For Women Men Waterproof College Gym Backpack Lightweight Small Travel Backpack Rucksack Casual Daypack Laptop Backpacks Hiking Backpack	Link	13.3748

When backpack as the product name

The shown vader scores are sum of the reviews it is , by which we can conclude that the higher vader score id ,being great the product is with positive appreciation given by users.



Product Vader Scores

	Product URL	Vader Scores Sum
Stanley IceFlow Stainless Steel Tumbler with Straw - Vacuum Insulated Water Bottle for Home, Office or Car - Reusable Cup with Straw Leakproof Flip - Cold for 12 Hours or Iced for 2 Days (Cream)	Link	2.8332
RTIC 40 oz Vacuum Insulated Bottle, Metal Stainless Steel Double Wall Insulation, BPA Free Reusable, Leak-Proof Thermos Flask for Water, Hot and Cold Drinks, Travel, Sports, Camping, Navy	Link	10.5494
Simple Modern 40 oz Tumbler with Handle and Straw Lid | Insulated Cup Reusable Stainless Steel Water Bottle Travel Mug Cupholder Friendly | Gifts for Women Men Him Her | Trek Collection | Mauve Me	Link	11.8626
YETI Yonder 1L/34 oz Water Bottle with Yonder Chug Cap, Charcoal	Link	11.2668
YETI Rambler 26 oz Bottle, Vacuum Insulated, Stainless Steel with Chug Cap	Link	12.4674
Trebo Water Bottle 64oz with Paracord Handle, Half Gallon Food-grade Double Wall Vacuum Stainless Steel Insulated Jug with Straw Spout Handle Lids, Leakproof Keep Cold & Hot, Ombre: Indigo/Black	Link	12.9814
RAYMYLO Insulated Water Bottle 64 oz, Triple Wall Vacuum Stainless Steel (Cold for 48 Hrs), Leak Proof & BPA-Free, Half Gallon Water Flask Jug with Paracord Handle & Straw Spout Lids, Indigo/Black	Link	7.4892
Coolflask 64 oz Water Bottle Insulated with Straw & 3 Lids, Large Metal Stainless Steel Water Jug Big Half Gallon Wide Mouth for Sports, Gym or Office, BPA-Free Keep Cold 48H Hot 24H, Magic Black	Link	7.2593
Takeya Actives Insulated Stainless Steel Water Bottle with Spout Lid, 32 Ounce, Midnight Blue	Link	3.9924
THIRST Quench H2.0 40 oz tumbler with handle and straw | Stainless Steel		

When water bottle as the product name given by the user

VII.WHY THIS APPROACH ?

Our innovative system offers a comprehensive solution to simplify the process of finding the perfect product on an ecommerce site. By leveraging a combination of machine learning algorithms and web scraping techniques, we address the challenges users face when navigating through a vast array of products online. Here's why our system stands out as the best solution:

Time Efficiency: Our system saves users precious time by automating the data collection and analysis process. Instead of manually sifting through countless products, users can quickly access top-ranked recommendations based on their preferences.

Accuracy and Relevance: We ensure accuracy in product recommendations by utilizing sophisticated clustering algorithms for pricing optimization. This ensures that users are presented with products that match their specific needs and budget.

Sentiment Analysis: Our system goes a step further by incorporating sentiment analysis using Hugging Face's pretrained NLP model and Vader scores. This allows users to gauge the emotional tone of product reviews, helping them make informed decisions based on the sentiments expressed by other customers.

Data Deduplication and Preprocessing: We implement data deduplication and preprocessing to provide clean and organized product information, reducing the chances of confusion or redundancy in the recommendations.

Transparency: The system not only provides product recommendations but also supplies users with URLs to the top-ranked products. This level of transparency enables users to verify the recommendations and explore further details if desired.

Adaptability: Our system can be easily adapted to work with various ecommerce sites, expanding its usability and relevance across a wide range of platforms.

Our system leverages ML technology to streamline the product selection process for users. By combining machine learning, web scraping, sentiment analysis, and data optimization techniques, we offer a powerful tool that saves time, enhances accuracy, and provides valuable insights into the world of online shopping. This makes our system the best choice for users seeking efficient and informed product recommendations on ecommerce platforms like Amazon.

Predictions that can be made using the described system for simplifying the process of finding the perfect product on an ecommerce site include:

Product Recommendations: The system can predict and recommend products that are most likely to meet a user's needs and preferences based on their input and the collected data, including product descriptions, review counts, overall ratings, and specific rating percentages.

Pricing Optimization: Through clustering algorithms, the system can predict optimal price ranges for products, helping users find products that offer the best value for their budget.

Customer Sentiment Analysis: The system can predict the sentiment of customer reviews using NLP models and provide insights into whether a product has positive or negative sentiment based on the reviews.

Product Availability: By scraping data from ecommerce sites, the system can predict whether a product is in stock or available for purchase.

Product Trends: Over time, the system can analyze data to identify trends in product popularity, thus helping users.

In conclusion, the system excels at real-time data scraping, robust analysis, and precise filtering, enabling it to select the optimal product that aligns perfectly with the unique requirements and specifications of any user.

VIII.FUTURE WORKS:

Gathering product information from Amazon via web scraping and subsequently identifying the most suitable product URL using ratings and sentiment analysis of reviews opens up numerous possibilities and promising avenues for future applications.

1)Competitor Analysis: You can track competitors' product ratings and reviews over time to gain insights into market trends and how they are perceived by customers.

2) Product Quality Assessment: Manufacturers and sellers can use this data to assess the quality of their products by analyzing customer feedback and making necessary improvements.

3)Customized Shopping Experiences: Leverage the data to offer personalized shopping experiences, suggesting products based on individual preferences and reviews they've given in the past.

4) Sentiment-Driven Marketing Campaigns: Develop marketing strategies based on sentiment analysis. For example, launch targeted advertising campaigns for products with overwhelmingly positive reviews and ratings.

5)Trend Forecasting: Analyze Amazon product reviews to identify emerging trends and consumer preferences. This information can be valuable for retailers and manufacturers planning their product lines.

6) Price Comparison Websites: You can use this data to build or enhance price comparison websites. Users often seek the best deals, and your analysis can help them find not only the cheapest products but also those with high ratings and positive sentiment in reviews.

7) Market Research: Analyzing sentiment from product reviews can provide insights into customer opinions and preferences. Companies can use this data to improve their products, marketing strategies, and customer service.

Whether for e-commerce platforms, market research, or personalized recommendations, this innovative approach is poised to make a lasting impact.

IX.CONCLUSION:

In today's digital age, where online shopping has become an integral part of our lives, the application of web scraping with automation tools like Selenium and Beautiful Soup in Python represents a game-changing solution. It enables the efficient collection of real-time data from the vast and dynamic landscape of e-commerce websites. This data, once acquired, undergoes a meticulous analysis and filtering process, resulting in the creation of a meticulously curated list of top-rated products. These ratings are determined not only by user reviews but also by VADER sentiment scores, which provide valuable insights into the overall satisfaction and sentiment associated with each product.

The true power of this application becomes apparent when we consider the arduous task it simplifies – product discovery. Online shoppers often find themselves inundated with an overwhelming array of choices. They must navigate through countless product listings, read reviews, compare features, and consider their budgets before making a decision. This process can be both time-consuming and mentally exhausting.

However, our application changes this paradigm. It empowers users to define their specific criteria and preferences, streamlining the process of identifying the ideal product. With just a few clicks, users can input their desired specifications, and in return, they receive a handpicked selection of product URLs accompanied by VADER sentiment scores. The significance of these scores cannot be overstated, as they serve as a reliable indicator of product quality and customer satisfaction. Higher VADER scores correlate with more positive reviews, ensuring that users are directed towards products that have garnered approval from their peers.

The time and effort saved by this streamlined approach are invaluable. Users no longer need to wade through endless lists of products, engaging in tiresome searches and comparisons. Instead, they can swiftly pinpoint products that not only meet their specifications but also align with their budgetary constraints. This newfound efficiency transforms the online shopping experience from a potentially frustrating and time-consuming endeavor into a pleasant and convenient one.

Furthermore, automation is the linchpin of this innovation. By automating the data collection and analysis process, our application ensures that users have access to real-time information. In the ever-evolving world of e-commerce, where products and reviews can change rapidly, this real-time advantage is a game-changer. Users can trust that the information they receive is up-to-date and reflective of the current market landscape.

Beyond the practical benefits, our application stands as a testament to the capabilities of web scraping and data analysis. It showcases how technology can be harnessed to simplify complex tasks, providing tangible benefits to users. Additionally, it underscores the importance of data-driven decision-making. In an age where information is abundant, making informed choices is paramount. Our application equips users with the tools they need to make those choices confidently, backed by data and sentiment analysis.

In conclusion, the application of web scraping, powered by automation through Selenium and Beautiful Soup in Python, is not merely a tool; it is a transformative force in the world of online shopping. It offers users a dynamic and efficient way to discover products that align with their preferences and budgets. It exemplifies the potential of technology to enhance our lives by simplifying complex processes. As we move forward in the digital age, applications like these will continue to redefine the way we interact with the online marketplace, making it smarter, more efficient, and more user-friendly.

X.REFERENCES:

1.“EFFICIENT SCRAPING OF DATA FROM WEBSITES USING SELENIUM”
Authors: [1]Shreya V. Dhoke student ,[2] Anupama D. Sakhare Asst Prof,[3] Satish J.Sharma Prof.
1,2,3 Dept of Electronics and Computer Science, Rakshtrasant Tukdoji Maharaj Nagpur
University,India . 2022 jetir, volume 9, issue 6,issn:2349-5162
https://www.jetir.org/papers/JETIRFM06063.pdf

2.“RoBERTa-GRU: A Hybrid Deep Learning Model for Enhanced Sentiment Analysis”
Authors:Kian Long Tan ,Chin Poo Lee, Kian Ming Lim ,Muiltimedia University,Malaysia
https://doi.org/10.3390/app13063915

3.“The k-means Algorithm: A Comprehensive Survey and Performance Evaluation”

Authors: Mohiuddin Ahmed, Raihan Seraj, Syed Mohammed Shamsul Islam
https://www.mdpi.com/2079-9292/9/8/1295

4.“Sentiment analysis of product reviews: A review”
Authors: T.K.Shivaprasad, Jyothi Shetty, NMAM Institute of Technology
https://ieeexplore.ieee.org/document/7975207

5.“A review on sentiment analysis and emotion detection from text”
Authors: Nandwani, Rupali Verma
https://link.springer.com/article/10.1007/s13278-021-00776-6#article-info

6.“VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text”
Authors:C.J.Hutto, Eric Gilbert ,Georgia Institute Of Technology
http://eegilbert.org/papers/icwsm14.vader.hutto.pdf

7.“An overview and comparison of free Python libraries for data mining and big data analysis”
Authors: Stacin, Jovic
https://ieeexplore.ieee.org/document/8757088

8. “An Automated Web Application Testing System”
Authors:Tarek M Mahmoud,Moheb Girgis,Bahgat A.Abudullatif,Alaa Zaki
https://www.researchgate.net/publication/270569315_An_Automated_Web_Application_Testing_System

9.”Natural Language Processing (Almost) from Scratch”
Authors: Ronan Collobert,Jason Weston,Leon Bottou, Michael Karlen ,Koray Kavukcuoglu,Pavel Kuksa
https://www.jmlr.org/papers/volume12/collobert11a/collobert11a.pdf

10.”A machine learning approach to product review disambiguation based on function, form and behavior classification”
Authors:Abhinav Singh, Conrad S. Tucker
https://www.sciencedirect.com/science/article/pii/S0167923617300477

11.”Deep Learning Approach of Product Evaluation Using Comment Analysis”
Authors: Syed Mudasar, Dr. Jasmeen Gill
https://doi.org/10.22214/ijraset.2021.39382

12.” Evaluation Of Product Reviews Using Deep Learning Classifier Models”
Authors: Lakshay Arora,Pallak Srivastava,Prayagraj, Ananda Kumar S
https://ieeexplore.ieee.org/document/9984463

13.” Machine learning for product choice prediction”
Authors: Josué Martínez-Garmendia
https://link.springer.com/article/10.1057/s41270-023-00217-7

14.”A Framework for the Design and Evaluation of Machine Learning Applications”
Authors: Kristian J. Hammond, Ryan Jenkins, Leilani H. Gilpin, Sarah Loehr
https://casmi.northwestern.edu/documents/evaluation-framework.pdf

15.” A Review on Machine Learning Strategies for Real-World Engineering Applications”
Authors: Rutvij H. Jhaveri,A. Revathi,Kadiyala Ramana, Roshani Raut,Rajesh Kumar Dhanaraj

https://www.hindawi.com/journals/misy/2022/1833507/