Paper Title

Avinash Shelke1, Akhilesh Dixit1, Nikhil Meshram1

Department Of Computer Engineering

Vishwakarma Institute of Information Technology Pune

***Abstract:***

***This project is a website that allows users to compare the prices of the same product from several online shopping websites. It also provides a sentiment analysis of the product to help users make informed decisions about their purchases. The website uses web scraping techniques to extract product information from ecommerce websites and display it on the website. It also performs sentiment analysis on customer reviews to indicate whether the product is worth buying or not. This website will provide a valuable service to users who want to compare prices and make informed decisions about their online purchases. It will also provide a useful tool for businesses that want to monitor the prices and customer sentiment of their products on different shopping websites.***

***Keywords:***

*Price comparison, Web scraping, Sentiment analysis, Natural language processing (NLP)*

1. **Introduction:**
2. **E-commerce product comparison**

When comparing a particular product on different ecommerce websites, such as Amazon and Flipkart, there are several factors to consider.

1. It is important to compare the prices of the product on

each website, as they may vary depending on the seller, promotions, or discounts available, and other factors. This can be done by searching for the product on both websites and comparing the prices listed.

1. Compare the shipping and delivery options available on each website. This includes the estimated delivery time, shipping cost, and whether the website offers free shipping. Some websites may also offer same-day or next-day delivery options, which can be a deciding factor for some buyers.
2. Consider the availability of customer reviews and ratings for the product on each website. Reviews and ratings can give valuable insights into the quality of the product, and can also help you identify any potential issues or drawbacks.

Overall, comparing a particular product on different ecommerce websites like Amazon and Flipkart involves considering several different factors, including price, customer reviews and ratings, and user experience. By carefully considering all these factors, an informed decision about where to buy the product that best meets the needs and preferences.

1. **Web Scraping**

Web scraping is the process of extracting data from websites automatically using a program or a script. Ecommerce websites are a common target for web scraping because they contain a large amount of data that can be useful for various purposes, such as price comparison, market research, and content analysis. To scrape product details from an Ecommerce website, you can follow these general steps:

1. Identify the target website and the product pages you want to scrape.
2. Inspect the HTML code of the product pages to identify the elements that contain the product details that need to be extracted, such as the product name, description, price, image, reviews, and ratings.
3. Create a web scraping program or a script using a programming language such as Python or JavaScript. Libraries such as Beautiful Soup, Scrapy, or Puppeteer can be used to simplify the scraping process.
4. Run the web scraping program or script and collect the product details from the target website. Issues such as anti-scraping measures, page navigation, and data formatting need to be handled.
5. Store the scraped data in a structured format such as CSV, JSON, or a database. Data visualization tools can be used to explore and analyse the scraped data.

It's important to note that web scraping can be a legally sensitive area, especially when scraping data from websites that have terms of use or copyright restrictions. Before scraping any website, the website's terms of use and their rules and policies must be considered. Additionally, some websites may block or restrict scraping activities, so it's important to use web scraping responsibly and ethically.

B1. Scraping Product Information

1. Amazon:

Input the name of the product from the user.

Figured out the product URL change according to the product name.

productUrl = "<https://www.amazon.in/s?k=>"+ product name

When constructing a URL for a product on an e-commerce website, whitespaces in the product name need to be replaced with some form of delimiter to ensure that the URL works correctly. Amazon has (+) sign as a delimiter.

There are many libraries available for web scraping in Python like Scrapy, Beautiful Soup, Requests, Selenium, PyQuery, LXML, Urllib. Beautiful Soup is a popular choice since it is easy to learn and use, powerful parsing capabilities, supports various parsers, a wide range of functionality, active community and documentation. requests and bs4 are to be imported for scraping the website.

The general flow of scraping via Beautiful Soup is:

1. Send a request to the URL and get the page content

response = requests.get(productUrl, headers)

In the request’s module, headers are used to send additional information along with an HTTP request. Headers contain metadata about the request or the client making the request. Some common headers used in requests include User-Agent, Accept, Authorization and Content-Type.

User-Agent is used to identify the client making the request, usually a web browser or a script. It includes information about the operating system and browser version.

1. Parse the page content using Beautiful Soup and the html.parser

soup= BeautifulSoup(response.content, 'html.parser')

1. Find the elements to extract data from using the find\_all() or find() method

element = soup.find('div', {'class': 'example-class'})

The orientation of the product details does not affect the class or data-hook value, which helps in standardizing the same scraping attributes for all product genres.

The functions declared for a particular detail scraping:

1. extractProductName(productUrl, response):

‘div’ = ‘span’

‘class’: ‘a-size-medium a-color-base a-text-normal’

1. extractProductUrls(productUrl,webpage):

‘div’ = ‘a’

‘class’: ‘a-link-normal s-underline-text s-underline-link-text s-link-style a-text-normal’

1. extractProductPrice(productUrl,webpage):

‘div’ = ‘span’

‘class’: ‘a-offscreen’

1. extractReviews(productUrl,webpage):

‘div’ = 'span'

‘class’: ‘a-size-base s-underline-text’

1. extractProductImage(productUrl,webpage):

‘div’ = ‘img’

‘class’: ‘s-image’

1. Extract the text or attribute values of the elements using the text or get() method

text = element.text

attribute\_value\_img = element.get('src')

attribute\_value\_link = element.get('href')

1. Flipkart:

Input the name of the product from the user.

Figured out the product URL change according to the product name.

productUrl = "<https://www.flipkart.com/search?q=>"+ product name. Flipkart also has (+) sign as a delimiter so the whitespaces in the product name need to be replaced with (+) to ensure that the URL works correctly requests and bs4 are to be imported for scraping the website.

The general flow of scraping via Beautiful Soup is:

1. Send a request to the URL and get the page content
2. response = requests.get(productUrl, headers)
3. Parse the page content using Beautiful Soup and the html.parser

soup= BeautifulSoup(response.content, 'html.parser')

Find the elements to extract data from using the find\_all() or find() method

element = soup.find('div', {'class': 'example-class'})

The orientation of product details changes the class value respectively. So, change in orientation of product details leads to change class value accordingly. The manual observations regarding the product orientation are stated below:

* 1. Vertical list orientation of the product details:

‘class’: ‘\_1fQZEK’

* 1. Horizontal list orientation (square) of the product details:

‘class’: ‘s1Q9rs’

* 1. Horizontal list orientation (rectangle) of the product details:

‘class’: ‘\_2UzuFa’

The functions declared for a particular detail scraping:

1. extractProductName(productUrl, response):

‘div’ = ‘a’

Depending on the orientation of product details:

‘class’: ‘\_1fQZEK’ or ‘class’: ‘s1Q9rs’ or ‘class’: ‘\_2UzuFa’

1. extractProductUrls(productUrl,webpage):

‘div’ = ‘a’

Depending on the orientation of product details:

‘class’: ‘\_1fQZEK’ or ‘class’: ‘s1Q9rs’ or ‘class’: ‘\_2UzuFa’

1. extractProductPrice(productUrl,webpage):

‘class’: ‘\_30jeq3’

1. extractReviews(productUrl,webpage):

‘div’ = ‘span’

class’: ‘\_2\_R\_DZ’

1. extractProductImage(productUrl,webpage):

‘div’ = ‘img’

‘class’: ‘\_396cs4’

1. Extract the text or attribute values of the elements using the text or get() method

text = element.text

attribute\_value\_img = element.get('src')

attribute\_value\_link = element.get('href')

B2. Scraping Product Review

1. Amazon

User clicked on the product details to get the sentiment analysis of the product. The productUrl associated with the product details is fetched and introduce changes to access the reviews page

For example:

Considered a product: Samsung galaxy f62

productUrl=‘[https://www.amazon.in/Samsung-Storage-6000mAh-Purchased Separately/dp/B09TWH8YHM/ref=sr\_1\_1\_sspa?crid=1O14H4QEVBZX0&keywords=samsung+galaxy+f62&qid=1682692033&sprefix=samsung+galaxy+f62%2Caps%2C216&sr=8-1-spons&sp\_csd=d2lkZ2V0TmFtZT1zcF9hdGY&psc=1’](https://www.amazon.in/Samsung-Storage-6000mAh-Purchased%20Separately/dp/B09TWH8YHM/ref=sr_1_1_sspa?crid=1O14H4QEVBZX0&keywords=samsung+galaxy+f62&qid=1682692033&sprefix=samsung+galaxy+f62%2Caps%2C216&sr=8-1-spons&sp_csd=d2lkZ2V0TmFtZT1zcF9hdGY&psc=1’)

reviewUrl=productUrl.replace(‘dp’,‘product-reviews’)+ "?pageNumber=" + str(pageNumber)

Functions declared for the product review scraping:

1. totalPages(reviewUrl):

It scrapes the review page and access the total number of reviews. Since each review page consists of 10 reviews the total number of review pages are calculated by dividing the total number of reviews by 10.

1. extractReviews(reviewUrl):

The function is intended to extract the product reviews.

1. Iterate over the totalPages and request each page to get the parsed data.
2. Send a request to the URL and get the page content

response = requests.get(productUrl, headers)

Headers contain metadata about the request or the client making the request. User-Agent includes information about the operating system and browser version.

Amazon AWS use Rate-based rule statement:

A rate-based rule tracks the rate of requests for each originating IP address, and triggers the rule action on IPs with rates that go over the limit that you set. You can use this type of rule to put a temporary block on requests from an IP address that's sending excessive requests. By default, this rule uses the IP address from the web request origin, but you can configure a different IP address source. (link)

Amazon does not allow multiple requests from a single IP address for web scraping because of which only one page with 10 reviews can be scraped for the sentiment analysis. Free proxy servers can be used for scraping of the reviews but the free proxies are well known and already blocked by the website.

1. Parse the page content using Beautiful Soup and the html.parser

soup = BeautifulSoup(response.content, 'html.parser')

1. Find the elements to extract data from using the find\_all() or find() method

element = soup.find('div', {'class': 'example-class'})

Functions declared for review scraping:

* 1. reviewTitle():

‘div’ = ‘a’

‘data-hook’: ‘review-title’

* 1. reviewBody():

‘div’ = ‘span’

‘data-hook’: ‘review-body’

1. Extract the text or attribute values of the elements using the text or get() method

text = element.text

attribute\_value\_img = element.get('src')

attribute\_value\_link = element.get('href')

1. Flipkart

User clicked on the product details to get the sentiment analysis of the product. The productUrl associated with the product details is fetched and introduce changes to access the reviews page.

For example:

Considered a product: Samsung galaxy f62

productUrl = ‘<https://www.flipkart.com/samsung-galaxy-f62-laser-green-128-gb/p/itm48722edba16c9?pid=MOBFZWSUZZ2Y7YYG&lid=LSTMOBFZWSUZZ2Y7YYGX2NE2S&marketplace=FLIPKART&q=samsung+f62&store=tyy%2F4io&srno=s_1_1&otracker=search&otracker1=search&fm=organic&iid=65b158ad-eed6-47e9-8935-b48d67223576.MOBFZWSUZZ2Y7YYG.SEARCH&ppt=dynamic&ppn=dynamic&ssid=amluvz202o0000001681964114830&qH=1686e6af328b9fbb>’

reviewUrl=productUrl.replace(‘/p/’,‘/product-reviews/’)+ "&page=" + str(pageNumber)

Functions declared for the product review scraping:

1. totalPages(reviewUrl):

It scrapes the review page and access the total number of reviews. Since each review page consists of 10 reviews the total number of review pages are calculated by dividing the total number of reviews by 10.

1. extractReviews(reviewUrl):

The function is intended to extract the product reviews.

1. Iterate over the totalPages and request each page to get the parsed data.
2. Send a request to the URL and get the page content response = requests.get(productUrl, headers)

Headers contain metadata about the request or the client making the request. User-Agent includes information about the operating system and browser version.

Flipkart allows multiple requests from a single IP address for web scraping. So, all reviews can be scraped but restricted to recent 1000 reviews because the time required to scrape the data is very high and the earliest review does not define the current status of the product.

1. Parse the page content using Beautiful Soup and the html.parser

soup = BeautifulSoup(response.content, 'html.parser')

1. Find the elements to extract data from using the find\_all() or find() method

element = soup.find('div', {'class': 'example-class'})

Functions declared for review scraping:

1. reviewTitle():

‘div’ = ‘p’

‘class’: ‘\_2-N8zT’

1. reviewBody():

‘div’ = ‘div’

‘class’: ‘t-ZTKy’

1. User clicked on the product details to get the sentiment analysis of the product. The productUrl associated with the product details is fetched and introduce changes to access the reviews page
2. **Sentiment Analysis on Review**
3. **Background Information**
   1. **Amazon and its product reviews**

Amazon product reviews are a valuable resource for both customers and sellers on Amazon's e-commerce platform. Product reviews are written by customers who have purchased and used a particular product, and they provide valuable information and feedback about the product's performance, quality, and other features.

The reviews are typically displayed in reverse chronological order, with the most recent reviews appearing first. Eachreview includes a star rating, written text, and sometimes photos or videos uploaded by the customer.

Customers can use Amazon product reviews to make informed purchasing decisions by reading about the experiences of other customers who have used the product. Product reviews can also help sellers to improve their products by identifying areas for improvement and responding to customer feedback.

Overall, Amazon product reviews are a valuable source of information for both customers and sellers, and they play an important role in the functioning of Amazon's e-commerce platform.

* 1. **Flipkart and its product reviews**

Flipkart is a popular e-commerce platform in India where customers can purchase a wide range of products online. One of the features of Flipkart is the ability for customers to leave product reviews. These reviews can provide valuable insights into the quality and usability of products for potential buyers.

Flipkart product reviews are typically written by customers who have already purchased and used the product. They can include ratings on a scale of 1 to 5 stars and a written description of the user's experience with the product. Reviews can cover a wide range of information, including product quality, delivery experience, customer service, and more.

* 1. **Sentiment Analysis**

Sentiment analysis is a natural language processing (NLP) technique used to determine the emotional tone or polarity of a piece of text. The goal of sentiment analysis is to classify the text as positive, negative, or neutral based on the language used in the text.

Sentiment analysis typically involves several steps, including text pre-processing, feature extraction, and classification. Text pre-processing involves cleaning the text data and removing irrelevant information such as stop words and punctuation. Feature extraction involves converting the text data into a numerical representation that can be used by machine learning algorithms. Classification involves using a machine learning model to predict the sentiment of the text based on the extracted features.

* 1. **Lexicon-Based and Machine Learning Techniques in Sentiment Analysis**

In the context of sentiment analysis, machine learning algorithms such as SVM (Support Vector Machine) and logistic regression can be used to classify text data into different sentiment categories such as positive, negative, or neutral. These algorithms learn to identify patterns and relationships in the data, and then use this knowledge to make predictions about the sentiment of new text data.

SVM is a popular machine learning algorithm used for classification tasks, and it works by finding a hyperplane in a high-dimensional space that best separates different classes of data. Logistic regression is another classification algorithm that estimates the probability of a binary outcome based on one or more input variables.

Lexicon-based approaches use pre-defined dictionaries or lexicons to identify and score the sentiment of words or phrases in a given text. VADER (Valence Aware Dictionary and sentiment Reasoner) and SentiWordNet are two popular lexicons used in sentiment analysis. VADER is a rule-based sentiment analysis tool that uses a dictionary of words and their valence scores to determine the sentiment of text. SentiWordNet is a lexical resource that assigns sentiment scores to words based on their synset (a set of synonyms that share a common meaning) and part-of-speech tags.

1. **Data**

Extract and pre-process data for sentiment analysis

* 1. **Collecting Amazon and Flipkart Reviews using real time web scraping**

Collecting Amazon and Flipkart reviews using real-time web scraping involves using an automated script or program to extract reviews from the product pages of Amazon and Flipkart websites. The process involves the following steps:

1. Identify the product pages on Amazon and Flipkart that contain the reviews for the products of interest.
2. Use web scraping tools such as Beautiful Soup, Scrapy, or Selenium to extract the HTML content of the review pages.
3. Parse the HTML content to extract the relevant information such as review text, review rating, reviewer name, and date of the review.
4. Store the extracted information in a structured format such as a database or CSV file.
   1. **Sampling Procedure**

The scraping of all reviews can be done but we restricted it to 1000 reviews for the products having more than 1000 reviews because of increase in computation time. The review title and review body section of reviews are scraped and stored in the database.

|  |  |  |
| --- | --- | --- |
| **Category** | **Data Type** | **Description** |
| Review Title | String | Brief description of review |
| Review Body | String | Detail description of review |

Table 1

1. **Text Pre-processing and TF-IDF Vectorization**
   1. **Text-Pre-processing using NLP methods**

Text pre-processing using NLP methods refers to the process of transforming raw text data into a format that can be easily analysed by machine learning models or other natural language processing tools. The process typically involves the following steps:

1. Tokenization: This involves breaking down the text into individual words or tokens. This can be done using tools such as NLTK or SpaCy.
2. Stop word removal: Stop words are common words that do not carry much meaning, such as "the", "and'', and "a". Removing stop words can help reduce the size of the dataset and improve the quality of the analysis.
3. Stemming and Lemmatization: Stemming involves reducing words to their base or root form, while lemmatization involves converting words to their dictionary or base form. Both techniques can help normalize the text data and reduce the dimensionality of the dataset.
4. Part-of-speech (POS) tagging: Labelling each word with its corresponding part-of-speech, such as noun, verb, adjective, or adverb. This can be done using tools such as NLTK or SpaCy.
5. Named entity recognition (NER): Identifying and classifying named entities such as people, places, and organizations in the text. This can be done using tools such as NLTK or SpaCy.
6. Spell checking and correction: Identifying and correcting spelling errors in the text. This can be done using tools such as PySpellChecker.
7. Text normalization: Transforming the text into a standardized format, such as converting all text to lowercase or removing punctuation.
   1. **Term Frequency Inverse Document Frequency (TF-IDF) Vectorization**

Term Frequency Inverse Document Frequency (TF-IDF) vectorization is a technique used to convert text data into a numerical representation that can be used for machine learning algorithms. The basic idea behind TF-IDF vectorization is to assign weights to each word in a document based on how frequently it appears in the document and how important it is in the corpus.

The TF-IDF vectorization process involves the following steps:

1. Tokenization: To break down the text into individual words or tokens.
2. Term frequency (TF) calculation: Calculate the number of times each word appears in the document. This is known as the term frequency (TF).
3. Inverse document frequency (IDF) calculation: The IDF value of a word is calculated as the logarithm of the total number of documents in the corpus divided by the number of documents that contain the word. The IDF value reflects how important a word is in the corpus.
4. TF-IDF calculation: The final step is to multiply the TF value of each word by its IDF value to obtain the TF-IDF weight of the word.

The TF-IDF model is constructed using two modules, TfidfVectorizer and TfidfTransformer, from the Python Scikit-learn library.

1. **Metrics and Modelling**

Sentiment Analysis of the reviews performed by using the two supervised learning algorithms logistic regression (LR) and Support Vector Machine (SVM) and two majorly used lexicons in NLP, Vader and SentiWordNet.

* 1. **Metrics of Binary Classification**

Binary classification is a type of supervised learning task where the goal is to predict whether a given data point belongs to one of two classes. In this type of classification, there are four possible outcomes:

* True Positive (TP): The model correctly predicts a positive sample.
* False Positive (FP): The model predicts a positive sample when the true label is negative.
* True Negative (TN): The model correctly predicts a negative sample.
* False Negative (FN): The model predicts a negative sample when the true label is positive.

There are several metrics used to evaluate the performance of a binary classification model, including:

1. Accuracy: the proportion of correctly classified samples among all the samples.
2. Precision: the proportion of correctly predicted positive samples among all predicted positive samples.
3. Recall: the proportion of correctly predicted positive samples among all true positive samples.
4. F1-score: a harmonic means of precision and recall.
5. Area Under the Receiver Operating Characteristic Curve (AUROC): a measure of how well the model distinguishes between positive and negative samples, regardless of the chosen classification threshold.
   1. **Supervised Machine Learning**

Python Scikit-learn library consists of supervised machine learning methods. The SGDClassifier for SVM model and LogisticRegression for LR model are used to classify the sentiment. Model is fitted using TF-IDF features of all words from reviews.In TF-IDF training model learns the frequency of words and vocabulary. From Amazon and Flipkart a collection of 10000 product reviews from 10 different products are extracted. The collection undergoestext pre-processing and TF-IDF feature extraction. Cross-validation of five-fold is used while training. On every iteration after shuffling, the fifth of the data is allocated for testing. Both the models are evaluated after

training on the basis of accuracy, precision, recall and F1-score.

* 1. **Lexicon-Based Learning**

VADER and SentiWordNet are fitted with pre-processed movie reviews, perform the sentence extraction, unescaping HTML escape sequences and expand contractions and then tokenize the tokens.

VADER has a scale of -4(most negative) to +4(most positive), the review sentiment is the summation of each sentiment score of the words in the review.

SentiWordNet additionally consists of POS tag for tokens, and rates the feature among positive, negative and neutral score. Document score is the summation of scores of individual words.

1. **Results**

Experimental results from four different techniques to classify sentiment of Amazon and Flipkart product reviews dataset. Experimental results of all evaluation parameters for all four classification algorithms are as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy  (%) | Precision  (%) | Recall  (%) | F1 Score  (%) |
| Support Vector Machine | 89 | 81 | 86 | 83 |
| Logistic Regression | 84 | 89 | 81 | 84 |
| VADER lexicon | 88 | 84 | 85 | 84 |
| SentiWordNet lexicon | 80 | 86 | 87 | 86 |

1. **References**
2. https://link.springer.com/article/10.1007/s40747-021-00436-4/tables/7