Aspect Based Sentiment Analysis: Visualizing User Reviews Based on Aspects

Nimisha Katyayani
Department of Computer Science
PES University
Bangalore, India
nimisha.katyayani@gmail.com

Krithi D
Department of Computer Science
PES University
Bangalore, India
deenadayalankrithi@gmail.com

***Abstract*— Aspect based sentiment analysis is an advanced text analysis technique. It involves breaking down the sentence to its key aspects and assigning sentiment to these specific aspects. This allows us to get a more accurate insight of our data and enables us to analyze large amounts of data. Analyzing customer reviews has always been an integral part of information processing. In this paper we use aspect based sentiment analysis on customer reviews to give the vendor a better insight of the user's opinion. This makes the company more customer centric.**

***Keywords*—** Aspects, Sentiment Analysis, Feature Extraction, Polarity Detection, Vectorization.

I. Introduction

The advancement in e-commerce has led to an upsurge of using technology as a tool and medium to shift processes from industrial based to a more information based economy. It is essential to understand how the consumer/customer conceives ideas about the product which is sold online.

There are numerous ways to find out customer perception towards products. The traditional ways of receiving feedback of the products were using surveys and forms, which is very time consuming and costly. Most websites have the option of customer rating, but this method is very ineffective as it would only give the overall opinion of the customer and each customer would have a different criterion to give it such a rating. Nowadays, few websites request you to do a survey to better understand the customer. These aren't as efficient as the respondents may not provide accurate information. In this digital era buyers can convey their opinions and judgment directly on e-commerce platforms and even on social media platforms. Twitter, Instagram, Amazon reviews are few of the interfaces a customer can use to share their thoughts on a product. These reviews, posts, comments are stored in the company's database but are not used as these large amounts of data can be very tedious to analyze.

In this paper we create our own data set of amazon reviews by web scraping. Amazon is the leading e-commerce company with over 280 billion dollars' net sales in the U.S alone in the year 2019. Amazon attracts a large number of consumers. They also provide a platform for their costumers to share their thoughts on different products. These review are unstructured and inconsistent making it very difficult to analyze. Many new technologies have been made to structure these large amounts of data so that the vendor can use this data to his advantage. One such technology we can use to process the data is Aspect based sentiment analysis. Aspect based sentiment analysis is a way of processing data by mining data / opinions, in our case it is in the form of reviews. It is a large array of data which is not being used efficiently. It can help vendors decide what the consumers want and what the demand is. Aspect based sentiment analysis can structure this large amount of underused data into an understandable format.

Our main goal in the paper is to provide a better understanding of each step of processing customer reviews in order to provide structure to this large array of data and to provide a more granular understand of a product. Ultimately providing the processed, structured information to the vendor. Each review is broken down to its key aspects and the sentiment of the customer towards each aspect. The top 5 aspects of each product is taken along with its sentiment. This information is provides to the vendor. The vendor can then analyze this data with ease.

After careful consideration we decided that data visualization of key aspects of the product would be the most useful way a vendor can utilize this information.

II. Literature Review

Over the last few years, several papers tried to improve estimation of unknown ratings of items by extracting useful information from user reviews and leveraging this information to achieve improvements in estimation. For example, the authors of a few papers found six aspects in the whole review and incorporated this information into a Matrix Factorization technique. There are various papers, which use an LDA-model approach towards the analysis of aspect based opinion mining, which also takes the rating of the products under consideration, and predicts a sentiment, which clearly justifies the review for that product, as given by the customer. There have been papers, whose authors have used the tensor flow approach towards the extraction of aspects, and thus, mining the opinions related to the respective aspects. The different libraries, as available in Python, help in the tensor flow approach. Our work is related to the aspect based recommender system, which is mainly, as the name suggests, recommends a product to the company, (which is the end user here), based on the particular aspect as chosen. We use the POS tags to extract the aspect terms, and then use the n-grams analysis to check for the adjectives describing these naming words. We obtain the polarity score for each of these describing words, and then calculate a few top aspects, for the product at hand, and keep these as the options, when the end user wants to check, what are the top few products, which are the best candidates, for the given feature. Once the aspect based sentiment analysis is done based on the reviews as written on the web page, we build a recommendation system, upon it, for the end user. The following sections describe the methodology of the procedure.

III. Sentiment Analysis

One way of structuring data is by using sentiment analysis. This method uses natural language processing, text analysis, computational linguistics and biometrics. It helps a business to understand the social sentiment of the product by categorizing it as positive, negative or neutral. It gives polarities for the sentiment words in a sentence and the net polarity is calculated and assigned to the sentence. First the data is gathered. Then the sentences are broken down into its component parts. The sentiment bearing words are found and sentiment score within the range of -1 and +1 is assigned to the sentiment words. These scores are combined and the net polarity is calculated. The polarity of the word can be taken from a sentiment library. It contains adjectives with their respective polarities. We use sentiment libraries mainly to maintain consistency and accuracy.

IV. Aspect Based Sentiment Analysis

Aspect based sentiment analysis is a more granular approach to sentiment analysis. A user review may contain multiple opinions about different characteristics of a product. In the regular sentiment analysis all the polarities are combined and the net polarity is taken. Whereas in aspect based sentiment analysis the sentence is first broken down into aspects and each aspect is given a polarity and the output would be the multiple sentiments found in the review.

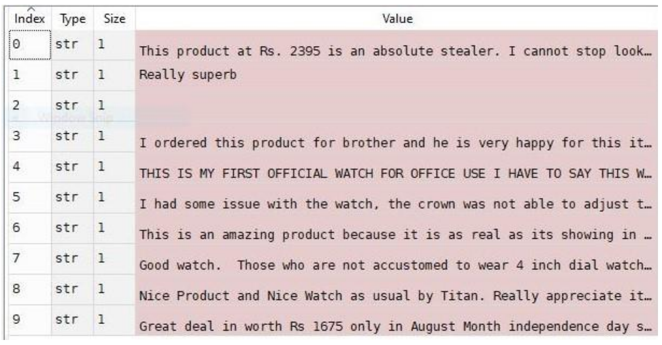
This is the quintuple definition of aspect based sentiment analysis,

S=(ei,aij,sijkl,hk,tl)
Ei – entity
Aij is an aspect of the entity
Sijkl defines the sentiment of the aspect
Ti is the time at which the sentiment is held
Hk is a sentiment holder

Take for example, “the phone’s camera quality is very good but the battery life is low”. These reviews address two features of the same phone. It has a positive review on the “camera quality” and a negative review on the “battery life”. Here “camera quality” and “battery life” are considered the aspects of the entity, the phone is the entity. “very good” and “low” is the sentiment of the aspect. The customer would be the sentiment holder which is identified by customer id. The aim is to find the most commonly mentioned aspects of the product. This along with the polarity of the aspect can be very crucial information for a vendor. The vendor can then focus on these aspects.

V. Text Analysis

Advancement in deep learning has made it possible for algorithms to analyze text. Text analysis also referred to as text mining, is the process of sorting unstructured text to gain valuable insight. This is important as it is difficult to keep track of tweets, customer reviews, and Instagram comments. This data can be gathered using data gathering. This data is used in text classification and text extraction models to extract machine readable facts from them. Text analysis supports multiple languages and dialects; we are gathering text of the English language.



Index	Type	Size	Value
0	str	1	This product at Rs. 2395 is an absolute stealer. I cannot stop look…
1	str	1	Really superb
2	str	1	
3	str	1	I ordered this product for brother and he is very happy for this it…
4	str	1	THIS IS MY FIRST OFFICIAL WATCH FOR OFFICE USE I HAVE TO SAY THIS W…
5	str	1	I had some issue with the watch, the crown was not able to adjust t…
6	str	1	This is an amazing product because it is as real as its showing in …
7	str	1	Good watch. Those who are not accustomed to wear 4 inch dial watch…
8	str	1	Nice Product and Nice Watch as usual by Titan. Really appreciate it…
9	str	1	Great deal in worth Rs 1675 only in August Month independence day s…

1: Reviews Extracted For a Product

Natural Language Processing combines concepts of linguistics, information engineering and artificial intelligence to make the interactions between a user and a computer even more versatile. It involves reading and “understanding” what a user is trying to say. Our task is to use different machine learning methods to break the natural text into a form understandable by the computer. This is done by following a series of steps.

NLTK, the natural language toolkit is a set of open source program modules which is used for natural language processing. NLTK has many modules which are discussed below.

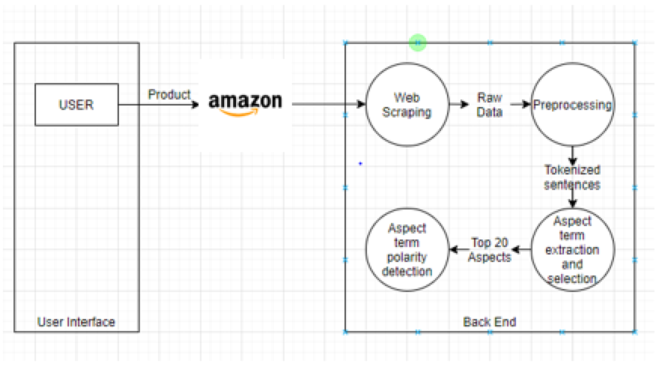


Figure 2: Design Architecture

VI. Method

A. Gathering data

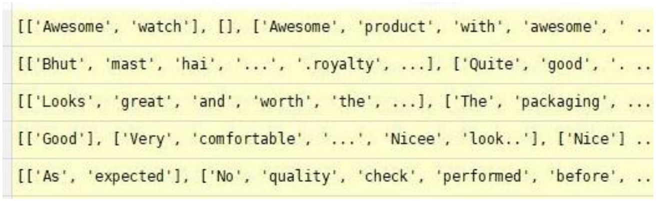
Web scraping: allows to extract large amounts of data from the web. By using request mechanisms, it allows the user to extract data from multiple websites. We have custom built a web scraper which is configured to scrape the amazon website. The data that we gain through web scraping is unstructured and is very difficult to analyze. It is too large for people to manually go through and too inconsistent for the computer to understand directly. This information has to be broken down to a computer readable format, which can be processed by the computer and later provided to humans in a more structured way for analysis.

The modules being used for web scraping are Requests and BeautifulSoup. Requests is a python module which enables us to make requests to a website. The websites URL and the search query can be provided and the "requests.get" makes a HTTP request to the web page.

BeautifulSoup is a python library which is used for extracting data from a html or a xml file. A custom web scraper must be built in order to access the data. Usually a section of a given website would follow the same formatting, or css styling. For example, the amazon website has a class which contains all the reviews. We specify this class and the tag and BeautifulSoup's module "findAll" would search the web page. The resulting output would be a list of reviews.

B. Preprocessing.

Preprocessing involves multiple steps to clean the data and save it in a more structured way so this data can be understood by the computer. The first step of preprocessing would be filtering out the noise in the data. The filtering out procedure starts with the splitting of the review, in the form of a list of sentences, which is known as sentence tokenization. A module in NLTK, called sent_tokenize, is helpful in creating tokens out of the sentences. "sent_tokenize" module in Python splits the review, based on full stops. It also splits the contractions, such that a word "don't" gets split into "don" and "'t".



```
[['Awesome', 'watch'], [], ['Awesome', 'product', 'with', 'awesome', ' ..
[['Bhut', 'mast', 'hai', '...', '.royalty', ...], ['Quite', 'good', '. ..
[['Looks', 'great', 'and', 'worth', 'the', ...], ['The', 'packaging', ...
[['Good'], ['Very', 'comfortable', '...', 'Nicee', 'look..'], ['Nice'] ..
[['As', 'expected'], ['No', 'quality', 'check', 'performed', 'before', ..
```

Figure 3 : tokenized list of words

Once a list of tokenized sentences is obtained, the next step is the correction of spelling mistakes. Since, it is a dataset consisting of only reviews, as written by the users on the website, we cannot expect the text to be formal, and very well-versed with the English language. The text is not expected to be written in perfect English, with the correct grammar and vocabulary. Hence, it is required to process the text in a way such that the spellings are all right, which will help in the later stages of textual analysis. Thus, we correct the spellings of the words, which are misspelt by the user. The segmented sentences are POS tagged using a dependency tree. We use the spacy dependency tree module to generate the dependency tree. We follow this approach rather than a word tokenization approach as finding the dependency between the noun and the adjective is much easier. It also handles steps like, removal of stop words (the removal of all the words, from the list, which do not contribute to the sentiment of the review, in any way, that is, words like pronouns, and prepositions.) Unnecessary punctuations, special characters and words are removed. Once we obtain a list of only tokens, consisting of only sentences, and no special characters or punctuations, we move on to the next step of preprocessing All this preprocessing of text, is followed by reducing the words into a basic form, which is called Lemmatization.

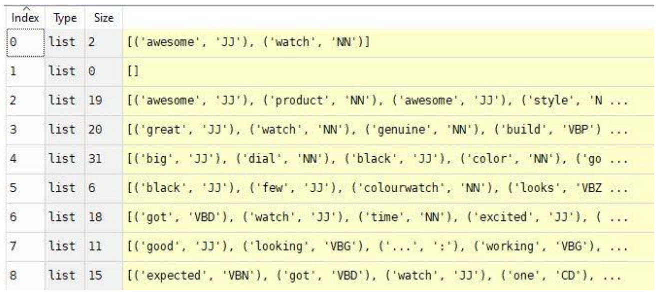
i. Sentence tokenization
ii. Correcting Spelling Mistakes
iii. Parts Of Speech tagging
iv. Generation of dependency tree
v. Lemmatizing
vi. Aspect Term Extraction and categorization
vii. Polarity Detection

i. Correcting Spelling Mistakes

The data may contain misspelled words or short forms. It is very difficult to predict these words with complete accuracy. Identification of the short forms or misspelled words is done using the TextBlob module as available in Python. It is mainly used for text analysis. It has multiple features such as noun extraction, POS tagging, Tokenization, etc. One such feature is spelling correcting. TextBlob.correct() method helps us find the correct spelling of a misspelled word.

Our data set also contains texts of different language written in English. We chose to ignore such reviews.

ii. Parts of speech tagging



Index	Type	Size	
0	list	2	[('awesome', 'JJ'), ('watch', 'NN')]
1	list	0	[]
2	list	19	[('awesome', 'JJ'), ('product', 'NN'), ('awesome', 'JJ'), ('style', 'N ...
3	list	20	[('great', 'JJ'), ('watch', 'NN'), ('genuine', 'NN'), ('build', 'VBP') ...
4	list	31	[('big', 'JJ'), ('dial', 'NN'), ('black', 'JJ'), ('color', 'NN'), ('go ...
5	list	6	[('black', 'JJ'), ('few', 'JJ'), ('colourwatch', 'NN'), ('looks', 'VBZ ...
6	list	18	[('got', 'VBD'), ('watch', 'JJ'), ('time', 'NN'), ('excited', 'JJ'), (...
7	list	11	[('good', 'JJ'), ('looking', 'VBG'), ('...', ':'), ('working', 'VBG'), ...
8	list	15	[('expected', 'VBN'), ('got', 'VBD'), ('watch', 'JJ'), ('one', 'CD'), ...

Figure 5: POS Tags

It is the process of identifying the verbs, nouns, adjectives and adverbs, i.e. determining the part of speech of the word. This makes it easier to find the aspect word in the later stages, the aspect words would be the word tagged as nouns and the sentiment would be the adjectives. Matching the adjective to the noun gives us the sentiment associated with the noun. This mapping is done using a dependency parser.

iii. Dependency tree is generated

We assume that the sentence is grammatically correct, this makes. The advantage of this approach is that it is based on the fact that the English language follows a standard structure.

The Aspect and sentiment pair is created using the dependency parser. It does this by connecting the words that could be considered as arguments of other words. Using a dependency tree allows us to map the right adjective to the aspect. Consider a review consisting of multiple sentences. Each sentence defines the aspect and its sentiment. It follows certain rules to generate the relationships.

iv. Lemmatization

Lemmatization is the process of bringing the word to their most basic form. It enables us to do morphological analysis of the words. The place where the user enters the search query, mainly uses the concept of lemmatization, which is nothing but the grouping of different forms of the same word into one set. By lemmatization, we reduce the number of words as obtained after the preprocessing techniques, and list down only the basic forms of all words. For example, the words, “running”, “ran”, “runs” will all be reduced to a basic form, called “run”, which helps in understanding the morphology as all the different forms of the word, basically have the same meaning.

v. Aspect dictionary

We create an aspect dictionary of each aspect. Different users may use different terms to describe the same aspect. We handle this by creating categories for every product. For every aspect term found in a sentence we try to categorize them into the predefined categories. This will eliminate inconsistencies in data.

For example,

- {this product is amazing}
- {best watch ever! I love it!}
- {this wristwatch is very good for its price}

Here, the words “watch”, “product”, “wrist watch”, “it” and “its” all means the same. This may cause inconsistencies in data. To avoid this, we can categories all these words under “watch”.

C. Aspect Term Extraction

Aspects mean, the features of the product in question. In our work, the product under consideration is “titan men’s watches”. Amazon uses similar structure for all its web pages, no matter what the product. Only the HTML tags and their respective ID and classes keep changing. Hence, for a different search query, the code remains the same, except the fact that the tags for extraction of the unique identification number (ASIN) and for the extraction of product names, need to be changed. Here, the search query is hard-coded, and hence the tags are used, according to what the query is.

For the product category, “Men’s watches”, there could be various features, or aspects, that the users could have an opinion about, such as, Dial Size, Weight, Battery, Look, and so on. The code is written in a way such that these aspect terms are extracted, and put alongside the words that describe these aspects. This is done using the concept of POS Tagging. The tokenized, preprocessed words get tagged with the part of speech they belong to, such as Nouns, Adjectives, Adverbs, etc. The nouns are parts of speech that are the aspects, since these are the naming words. The adjectives are the terms that describe these aspects. This is the principle we use in the aspect term extraction step.

The extracted nouns are then used with word2vec features to compute the similarities between them. Word2vec processes data by vectorizing the words in vector space using neural word embedding. Neural word embedding allows us to represent words mathematically in an n – dimension vector space. The distance of similar words would be smaller than that of dissimilar words. These words are then grouped together under aspect categories. This enables us to remove the variation in the data. The sentiment of each aspect in each review is mapped. Then the polarity can be calculated.

D. Polarity Detection

Once the dictionary is formed as above, we use the Textblob library available in python to estimate whether the adjectives as used to describe the respective nouns are positive, negative, or neutral. Once the polarity for each of the aspects is calculated, we go ahead and thus, filter out the top few aspects for each of the products, and thus train our model as such.

Any new entry will be analyzed like the above procedure, and thus, will give a clear idea to the company, about what are the important aspects for the given product under consideration, and thus can work upon building the respective sales and marketing, for the product.

E. Sentiment Score

There are three sentiments that we are mainly aiming at, Positive. Negative, and Neutral. Once we obtain a data structure of the form,
{Product : {Aspect:SentimentScore}}, for every product, we filter the structure by taking the top few aspects for a particular product, decided by the sentiment score, and thus, train the model, as per the entire procedure as aforementioned.

F. Visualization

For example, our dataset is Titan watches, the end user or vendor would be Titan. If sturdiness is the aspect under consideration, multiple users would have commented on the durability of the watch. We can list the different products (from best to worst) according to the user opinion of the aspect sturdiness.

G. Optimization of Code

i. Selenium

Selenium WebDriver drives browser natively, as a real user would, either locally or on remote machines. Without selenium when the code was run, whenever the internet connection was broken, or had a slow response, we faced a problem, where the reviews weren't getting extracted, even though the response code was that of success. Figuring the solution to this whole problem took quite some time, but the usage of selenium web driver helped the situation.

Selenium is an open source tool which automates web browsing. It helps in controlling the way Chrome works, by automating the entire process of scraping the data. It runs the python code, on the browser. Since the web browser used here is Google Chrome, the chrome driver executable file needs to be downloaded, and the respective path as to where the executable file is stored, needs to be put into where we are calling the selenium module.

The code excerpt as used in the application:

```
cookie={}
def getAmazonSearch(search_query):
        url="https://www.amazon.in/s?k="+
search_query
         driver =
webdriver.Chrome(r"C:/Program Files
(x86)/Google/Chrome/Application/Driver/c
hromedriver.exe")
        driver.implicitly_wait(10)
        driver.get(url)
   #print(url)
        page=requests.get(url,cookies=co
okie,headers=header)

        if page.status_code==200:
     #print (driver.page_source)
                return driver.page_source
         else:
                return "Error"
```

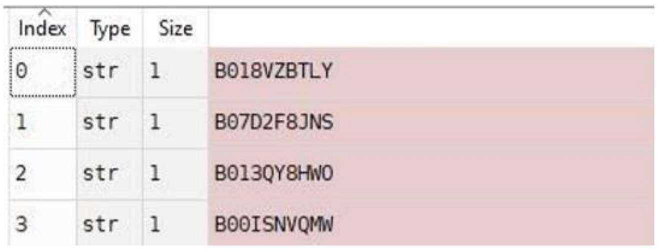
For other browsers, other driver executable files could be downloaded, as per the software requirements of the system. Using selenium, the code works just fine, and gives us the details, as per the tag entered for the beautiful soup library to extract the section of the web page.

ii. Threading

As the code extracts large amounts of data from the web it takes a lot of time. To decrease the response time, we use threading. Thread Pools enables us to run up to 10 links concurrently.

Code Excerpt:

```
threadPoolResponses1 = []       #collect
the threadpool responses
# A thread pool basically spawns the given
number of processes on the called function
# So the "Searchasin" function can be
called concurrently with 10 links at the
same time
# Since each thread in the pool has
separate memory space, to collect all the
responses, use "pool.imap_unordered"
# "pool.imap_unordered" takes the partial
function and an iterable which can be used
to iterate the data
 print("data_as_in", data_asin)
 func = partial(Searchasin, data_asin)
 with Pool(processes=15) as pool:
        for result in
pool.imap_unordered(func,
range(len(data_asin))):
```



Index	Type	Size	
0	str	1	B018VZBTLY
1	str	1	B07D2F8JNS
2	str	1	B013QY8HWO
3	str	1	B00ISNVQMW

Figure 6: The ASIN tags as obtained by the above code

This decreased the time to complete the web scraping significantly. Approximately 92.11% decrease in the time to complete web scraping.

ACKNOWLEDGMENT

First and foremost, we would like to thank our research supervisor Prof. Ashiwini M Joshi without whose assistance and dedicated involvement in every step throughout the process, this project would have never been accomplished. . She inspired us greatly to work in this project. Her willingness to motivate us contributed tremendously to our project. We also would like to thank her for showing us some example that related to the topic of our project. We would like to thank you very much for your support and understanding. We would also like to thank PES UNIVERSIY for providing us with a good environment and facilities to complete this project.

REFERENCES

[1] Soujanya Poria, Erik Cambria, Lun-Wei Ku Chen Gui Alexander Gelbukh "A rule-BasedApproachtoAspectExtraction from Product reviews".

[2] Qiming Du, Dinghan Zhu and Wenjing Duan "Recommendation System with Aspect-Based Sentiment Analysis"

[3] Konstantin Bauman, Bing Liu, Alexander Tuzhilin "Aspect Based Recommendations: Recommending Items with the Most Valuable Aspect Based on User Reviews"

[4] Ronen Feldman "Techniques and Applications for Sentiment Analysis

[5] Maria Pontikil, Dimitrios Galanis, John Pavlopoulos, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar “SemEval-2014 Task 4: Aspect Based Sentiment Analysi