Kozari Georgia

Design and development of a system based on short questions for retrieving relevant documents that express opinion.

MSc Dissertation

Supervisor: Koloniari G.

Thessaloniki, October 2019
Applied Informatics, University of Macedonia

DESIGN AND DEVELOPMENT OF A SYSTEM BASED ON SHORT QUESTIONS FOR RETRIEVING RELEVANT DOCUMENTS THAT EXPRESS OPINION.

Κοζάρη Γεωργία

Απόφοιτη του Τμήματος Μαθηματικών του Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης, 2017.

Διπλωματική Εργασία

υποβαλλόμενη για τη μερική εκπλήρωση των απαιτήσεων του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΤΙΤΛΟΥ ΣΠΟΥΔΩΝ ΣΤΗΝ ΕΦΑΡΜΟΣΜΕΝΗ ΠΛΗΡΟΦΟΡΙΚΗ

Επιβλέπουσα Καθηγήτρια

Κολωνιάρη Γεωργία

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή στις 31/10/2019

Κολωνιάρη Γεωργία	Ευαγγελίδης Γεώργιος	Βεργίδης Κωνσταντίνος
....	....	....

Κοζάρη Γεωργία

....

Abstract

Opinion mining is an interesting area of research because of its applications in various fields. Collecting opinions of people about products or social events and problems through the Web is becoming increasingly popular every day. The views of users are beneficial regarding both the public and stakeholders in terms of decision-making. Opinion mining is a way to retrieve information through search engines, Web blogs and social networks. Summarizing information manually constitutes a time consuming process due to the huge number of reviews in the form of unstructured text. Hence, effective and efficient computational methods in terms of mining and summarizing reviews from corpuses and web documents, are of critical importance.

This study aims at designing and implementing a system which will consist of an ensemble of different documents which express positive, negative or even a neutral view about an issue, a product or a person. Thus, the above mentioned system will generate results, ranked by certain criteria (e.g relevance, date, reliability) and corresponding to users' query. The latter will be in the form of filters, keywords or even the combination of them.

More specifically, many different methods of pre-processing will be used for the summarizing of reviews and user's query. Also, extraction of aspects and, in case of their detection to the user's query, expansion of them based on thesauri will be two main procedures of the system. Finally, the conduction of a variety of experiments while the use of the statistical measure Term Frequency-Inverse Document Frequency were essential for the evaluation of the system.

Keywords: Opinion mining, information retrieval, sentimental analysis, thesaurus, aspect extraction, query expansion.

Acknowledgments

I would like to thank my supervisor Dr.Koloniari Georgia of Applied Informatics at University of Macedonia, for the truly valuable recommendations and guidance throughout this year.

Public opinion alone can keep a society pure and healthy.

Mahatma Gandhi

Contents

List of Figures

List of Tables

Chapter 1

Introduction

How many times have you wondered that most of human decisions and therefore behaviors and activities are influenced by public opinion? Glancing over everyday life it is apparent that the beliefs, perceptions and the choices we make, are, to a considerable degree, influenced by how others see and evaluate the world. We buy a product, we book a hotel, we formulate our personal style and personality and we shape our opinion for others once reading relevant reviews or posts. What is more, this is not only true for individuals but also for organizations and businesses.

Reflecting on the last years, the proliferation of social media (e.g Twitter, Facebook and Instagram), forum discussions and blogs generate a huge amount of data. Through daily social media services, forums and news aggregators a great amount of views is exchanged. This generated data albeit a "goldmine" of information, does not reveal, in the first instance, every aspect of individual opinion as an aggregate. Hence, in order to identify individual opinions that comprise the core elements of aggregate data, processes that enable the mining of the former shall be employed.

To this end, though, cost-effectiveness and time-efficiency of respective opinion mining processes, are of crucial importance given the vast amount of data. Therefore, this dissertation aims to provide a cost-effective information retrieval process with enhanced time-efficiency but notably, without inflicting the relevancy between individual inquiries and respective outputs.

1.1 Opinion mining and the enquiry of information retrieval systems

“Opinion mining (OM) is a procedure used to extract opinion from text ” according to [**Khan et al.**, **2014**]. As the Internet and Web technologies continue to grow and expand, the space and scope in the area of information is also expanding and so OM is the way to retrieve this useful information through search engines, web blogs and social networks. Thus, “OM is a recent discipline at the crossroads of information retrieval, text mining and computational linguistics which tries to detect the opinions expressed in natural language texts” as [**Pang and Lee**, **2008**] referred, and its tasks involve opinion, target and source identification, opinion classification and opinion summarization. This is the reason why techniques from the field of Natural Language Processing (NLP), Information Retrieval (IR) and text mining are necessary.

In recent years, we have witnessed how opinionated posts, comments, reviews or tweets on social media, forum discussions, blogs and microblogs have helped reshape business and sway public opinion, profoundly impacting our social and political lives. However, the main concern remains and it is how to automatically identify opinion components about an entity from a huge volume of unstructured text and summarize them. The answer to the concern above comes from the field of the Information Retrieval (IR).

According to [**Manning et al.**, **2009**],“Infromation retrieval did not began with the Web. In response, to various challenges of providing information access, the field of information retrieval evolved to give principled approaches to searching various forms of content. The field began with scientific publications and library records”. However, IR soon spread to many forms of content, particularly those of information professionals, like journalists, businessmen, lawyers and doctors and has moved from being a primarily academic discipline to being the basis underlying most people’s preferred means of information access.

Definition 1.1.1 (Information Retrieval) : “Information Retrieval is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers)” [**Manning et al.**, **2009**] .

As [**Manning et al.**, **2009**] referred, another semantic part of IR is the efectiveness of an information retrieval system. In order to measure ad hoc information retrieval efectiveness in the standard way, we need a test collection consisting of three things:

- A document collection.
- A test suite of information needs, expressible as queries.
- A set of relevance judgments, standardly a binary assesment of either relevant or nonrelevant for each query-document pair.

Nevertheless, the most interesting part of IR is the new challenges and the motivation of researchers to look for intelligent information retrieval systems that search and/or filter information automatically based on some higher level of understanding, as [**Ravindra and Poonam**, **2012**] have referred.

Intelligent information retrieval—finding information truly relevant to a user's need—has become increasingly important because of the dramatically growing availability of online documents. Addressing this task requires synergy between information retrieval techniques and Artificial Intelligence (AI) research because automatic learning and prediction about the relevance of a document's content to the user's information need are crucial for satisfactory solutions. This special issue presents a set of creative approaches to new and important challenges in a variety of applications, including detection and tracking of novel events from news stories, automatic assignment of subject categories to articles, customized routing of e-mail messages, search and navigation through the World Wide Web, and indexing and retrieval of multimedia documents and they are presented in [**Yang and Pedersen**].

1.2 Dissertation's purpose

The purpose of this dissertation is the design and development of an intelligent information retrieval system. This system will be based on short questions-keywords and its aim will be to retrieve relevant documents that express opinion. Thus, the above system will generate results, ranked by certain criteria (e.g relevance) and corresponding to user's query. The latter will be in the form of filters, keywords or even the combination of them. In other words, like the miner when digs, he or she is searching for gold or other valuable objects, our system when searching in huge amounts of data aims to discover and extract only useful information, that is, the opinion that fits with the user's query.

One major challenge we have to face is that our system has to be cost-effective, time-efficient and flexible on the different domains of the data. So, a big variety of preprocessing methods of data and user's query have been used, while the use of procedure of extracting aspect words which gather the most significant part of the information of the data was necessary.

In addition, the significant method of the expansion of the user's query in synonyms, hypernyms, and hyponyms by using thesauruses, while the calculation of term frequency-inverse document frequency score in order to find the most relevant documents were essential for the design and evaluation of the system, respectively. All of these concepts, however, will be discussed in the next chapters, which are essential in order to understand the design, implementation and purpose of the system.

1.3 Dissertation's outline

The introductory part of the dissertation (Chapter 1) deals with the importance of creating and using information retrieval systems for opinion mining, and the need to develop these systems into intelligent information retrieval machines. It also contains the purpose of my dissertation which is to create such an intelligent information retrieval engine.

The next chapter (Chapter 2) contains the definitions of OM and many relevant parts of it (e.g opinion, opinion holder, the sentiment of opinion, applications of OM, the levels of sentiment analysis, etc.). Moreover, it includes a complete description of the preprocessing methods and the applications of thesauruses in OM. Finishing this chapter of the dissertation the reader also can find information about the statistical measure Term Frequency-Inverse Document Frequency (TF-IDF).

Chapter 3 contains all the methods we used for the designing and development of our information retrieval system and we summarized all of them into a figure (Figure 3.3) at the beginning of the chapter. Then, the first section of the chapter is referred to the procedure of the pre-processing of our data as well as to this of the user's query while the second one contains this of aspects' extraction technique. In the last part of the chapter, after the analysis of the expansion of the top-aspects and the query that anyone can insert to the system, we also analyzed how we can find the most relevant documents of our data to this query by using the score of TF-IDF.

The chapter of evaluation (Chapter 4) is the most interesting part of my dissertation as it contains the examination of the system that has been created. Speciffically, data that have been used, as well as, the experiments that have been conducted and the results of them are described in detail. Also, after the table of results of each experiment procedure there is a part of discussion of them where some remarks that have emerged are highlighted.

Finally, chapter 5 contains the conclusions that were drawn up during my dissertation and some suggestions for future research.

Chapter 2

Literature survey of relevant technological concepts

The term opinion mining (OM) is a core-term to which many concepts such as subjectivity detection, sentiment analysis, sentiment polarity, etc. are associated. However, this causes confusion between the concepts, as these terms often refer to each other in the literature. Thus, there are differences between them, that have to be noted. Also, as mentioned in the previous chapter semantic parts of OM, like preprocessing methods and thesaurus, have to be explained.

Therefore, this chapter will give a more accurate definition of OM and related terms like sentiment and opinion. The most significant applications and the different levels of sentiment analysis will also referred. In addition, will be recording the main methods of text's preprocessing which are essential for opinion detection and summarization. A particular analysis will also be made of the thesaurus and how synsets of synonyms, hypernyms, and hyponyms help to reduce the words of a text or extend it according to the occasion and requirements of the research. Finally, we will define the numerical statistic Term Frequency-Inverse Document Frequency (TF-IDF), that is intended to reflect how important a word is to a document in a collection or corpus.

2.1 Opinion Mining

In recent years there has been a great deal of confusion if opinion mining and sentiment analysis are two common words that can be interchanged in a text without changing their meaning. There are so many opinions about these two terms and as their fields of applications are quite mutual, it makes sense that they have many similarities and it is difficult to completely separate them. The term sentiment analysis first appeared in [**Nasukawa and Yi**, **2003**], and the term opinion mining first appeared in [**Dave et al.**, **2003**]. However, research on sentiment analysis and opinion mining began earlier as it is referred in [**Wiebe**, **2000; Das and Chen**, **2001; Morinaga et al.**, **2002; Pang et al.**, **2002; Turney**, **2002**]. Even earlier related work includes interpretation of metaphors; extraction of sentiment adjectives; affective computing; and analysis of subjectivity, viewpoints, and affects according to many different sources such as [**Wiebe**, **1990**, **1994; Hearst**, **1992; Hatzivassiloglou and McKeown**, **1997; Wiebe et al.**, **1999**].

In Merriam-Webster's dictionary, sentiment is defined as an attitude, thought, or judgment prompted by feeling, whereas opinion is defined as a view, judgment, or appraisal formed in the mind about a particular matter. The difference is quite subtle, and each contains some elements of the other. The definitions indicate that an opinion is more of a person's concrete view about something, whereas a sentiment is more of a feeling according to [**Liu**, **2015**].

In my dissertation, I use terms sentiment analysis and opinion mining interchangeable. Although most of the time I prefer the second one as it more accurately expresses my view. So it is time to give the precise definitions of the terms opinion mining, opinion and its components.

2.1.1 Definitions

Definition 2.1.1.1 (Opinion Mining) : Given a set of evaluative text documents D that contain opinions (or sentiments) about an entity (e.g item/topic/person/product or service), opinion mining aims to extract aspects (e.g properties or attributes) of the entity that have been commented on in each document

$$\mathrm{d} \in \mathrm{D}$$

and to determine whether the comments are positive, negative or neutral as [**Bakhatawar and Farouque**, **2012**] defined it.

Definition 2.1.1.2 (Opinion) : "An opinion is a quintuple,
(h, p, a, s, t) where h is the opinion holder, p is the entity (or purpose) of the opinion, a is the target aspect of the entity p on which the opinion has been given, s is the sentiment of the opinion on aspect a of entity p, and t is the opinion posting time; s can be positive, negative or neutral, or a rating (e.g, 1-5 stars)" as [**Liu**, **2015**] has referred.

Below we will define the three main components of opinion :

Definition 2.1.1.2.a (Opinion holder) : The opinion holder is often the author of the text in which the opinion is expressed. However, this is not always the same, as it is possible for the author to quote the opinion of a third person. In addition, the opinion holder may not be a natural person, but an organization or group of people. For example, it might be a newspaper, or a political party, which expresses its opinion on a public event, such as the result of a referendum, or even a company that wants to enhance its platform of consumers' reviews.

Definition 2.1.1.2.b (Object of Opinion) : The object of opinion is the entity, or aspect of the entity, for which the opinion is expressed. It can also be represented as a hierarchy, in which one aspect is in turn one that is characteristic of another aspect. For example, an entity could be a car, for which one aspect is expressed(e.g 'reliability'), which is similar to other aspects such as 'engine durability' or 'plastic durability'.

" This definition describes an entity hierarchy based on the part-of relation" according to [**Liu**, **2015**]. The root node is the name of the entity (e.g the name/brand of the car). All the other nodes are parts and subparts. Thus, opinion can be expressed on the root-node as well as on the aspect-parts or subparts.

In the research literature, entities are also called objects, and attributes are also called features (as in product features) [**Hu and Liu**, **2004; Liu**, **2010**]. We choose not to use the terms object and feature in this dissertation because "object" can be confused with the term object used in grammar, and "feature" can be confused with feature used in machine learning to mean a data attribute. In recent years, the term aspect has become popular and covers both part and attribute.

Definition 2.1.1.2.c (Sentiment of Opinion) : Sentiment is the most important attribute that defines an opinion and one that has been studied more than any other in the context of OM. It is about the set of emotions, ideas and attitudes that come from an opinion or a text.

It is represented as a triple, **(y, o, i)** , where y is the type of the sentiment, o is the orientation of the sentiment, and i is the intensity of the sentiment.

Type of sentiment, y :

Sentiment can be classified into several types. There are linguistic-based, psycology-based , and consumer research-based classifications. In my research, I choose to use a consumer research-based classification because I feel it is simple and easy to use in practice. Consumer reseach classifies sentiment broadly into two categories : 1) rational sentiment and 2) emotional sentiment according to [**Chaudhuri**, **2006**].

Orientation of sentiment, o :

It can be positive, negative, or neutral. Neutral usually means the absence of sentiment or no sentiment. Sentiment orientation is also called polarity, semantic orientation, or valence in the research literature.

Intensity of sentiment, i :

It determines how strong the sentiment is expressed. The gravity of a sentiment can be given either by words such as ”excellent” instead of ”good”, or by quantitative adverbs such as ”very much”, or by affirmative adverbs such as ”minimally”.

2.1.2 Applications of opinion mining

Opinion mining have spread to almost every possible domain, from consumer products, health care, tourism, hospitality and financial services to social events, political elections and goverment agencies. Hence, acquiring and analyzing consumers' and public views and reviews nowadays is a huge business for marketing, public relations and political campaign firms. That is why individuals, organizations, businesses and goverment agencies are increasingly using the content of views or reviews for decision making. According to [**Pang and Lee**, **2008**], opinion mining, though an intellectually difficult problem, is extremely useful in practical applications. We analyze the main applications below;

Shopping

“ The most popular use of opinion mining is for the decision support system to consumers. Consumers are actively involved in shopping over the world” according to [**Kalaria and Prajapati**, **2016**]. In addition, online shopping has become increasingly popular and useful because of the convenience and time savings it offers. Thus, before the consumer makes an online purchase, he or she is informed about the features of different products or services and compare with each other. These features may relate to support, delivery time and shipping charges and can be found in the comments or posts of their current customers. In fact, many popular websites like Amazon, Flipkart and Snapdeal allow or sometimes require consumers to express their views or reviews about their products or services.

Entertainment

Another important opinion mining application domain is that of entertainment, which involves both songs, movies, concerts and theaters as well as travel, restaurants and bars. For example, movie and home TV viewers can easily access the opinion on recent releases and popular movies and programs by IMDB which is an internet movie database that provides us online score and quick reviews about a movie, a serie or a TV program. A proportional song/video database is Youtube where you can find a huge volume of songs and videos with their relevant comments that in addition to user's fun can be useful information as they may contain comments about the artist or the type of a song or the methods and services mentioned in the videos. Moreover, there are special platforms or even the event website where the public but also critics can express their opinion about events like a concert or a theater. Finally, service websites, like TripAdvisor, are suitable for searching opinions for the ideal hotel for your trip but also for the ideal restaurant or bar.

Business

The applications of OM in business cannot be overlooked. It is important for a business to know the opinion of its customers about its products or services so that they can improve and design its strategy. It is also important to know the opinion of the world about its competitors and find out what its comparative advantages are and in which areas it is lagging behind in order to improve its competitiveness. Getting x% negative or positive reviews on a certain product doesn't make much sense if you don't have a y% metric to compare it with. Knowing the sentiment data of your competitors gives you the opportunity as well as the incentive to perk up your performance. OM in businesses can be very helpful in predicting the customer trends. Once you get acquainted with the current customer trends, strategies can easily be developed to capitalize on them. And eventually, gain a leading edge in the competition. " Thus, companies can modify their products according to customers' opinions in a better and faster way and they can establish better customer relationship by giving them exactly what they need" [**Seerat and Azam**, **2012**]. To achieve these goals, it is extremely useful to utilize reviews in online stores (such as Amazon) but also in social media (Twitter, Facebook and Instagram). This data is relatively easy to collect, but it is difficult to extract useful information from it.

Goverment

"In addition to business interests, applications are also widespread in government agencies. Internally, agencies monitor social media to discover public sentiments and citizen concerns" according to [**Liu**, **2015**]. In the past, when governments wanted to know what was happening in other countries, they monitored the traditional news media, for example, newspapers, radio, and TV, in these countries, and even sent spies to these countries to collect such information. However, this was an expensive and time-consuming procedure. Nowadays, as many commercial social media monitoring tools are available, goverments easily obtain public opinions about their policies and measure the pulses of other nations simply by monitoring the main social media sites of these countries.

Politics

Social media enables people to participate in political discussions on issues that affect their lives. In politics, public opinion is crucial for parties or candidates, as it greatly influences their strategy. OM has two main policy applications, 1) real-time polling [**O'Connor et al.**, **2010; Maynard et al.**, **2011; Stieglitz et al.**, **2012; Wang et al.**, **2012; Zhou et al.**, **2013**] and consequently 2) the prediction of the election results [**Ceron et al.**, **2014**].

Conducting polls is an expensive and time-consuming process. Careful design of questionnaires is required as useful questions may be omitted or some of them may be formulated incorrectly. In addition, a major problem is the length of the polls. For example, if an important issue arises (natural disaster, political criticism, political scandal, etc.) every now and then a company undertakes to conduct a poll. The purpose of the poll is to analyze the public opinion on how the political situation is managed by each political body. The poll lasts a few days, but during this time it is likely that developments have occurred that will not be reflected in the responses of those surveyed at the beginning of the poll. Therefore, the results will not be the most representative.

These problems can be solved by conducting public opinion polls in real-time. This is achieved by using OM systems, analyzing the wording that is formulated on social media. Initially, this type of polling is passive and once data is collected (messages, comments, articles, etc.) there is no limit to the number and type of queries that will be executed. Thus, it is possible to execute a new query in the data and stakeholders have immediate results at their disposal, at a later time. This means that a party or candidate may be fully aware of the public's pressures and thus make more informed decisions.

Other applications

Although OM can be applied to the social and business sectors, researchers are also making an effort to effectively employ it in other important areas, e.g health, education etc. [**Goeuriot et al.**, **2011**] proposed social media sites where people can post information about their disease and treatment for the purpose of mining disease and treatment information. Finally, according to [**Abulaish et al.**, **2009; Das et al.**, **2001; Feldman et al.**, **2007; Kessler et al.**, **2010; Lin and Chao**, **2010; Zhuang et al.**, **2006**], OM is being applied in several commercial areas such as tourism, automobile purchasing and electronical game or movie reviews as well as in various political arenas such as public administration, strategic planning, marketing etc.

The aforementioned works represent only a small sample of OM applications. Various surveys have been conducted regarding the existing works and the potential applications of OM in practical life, thus indicating the importance of opinion mining [**Pang and Lee**, **2008; Tang et al.**, **2009; Tsytsarau and Palpanas**, **2011**].

2.1.3 Levels of sentiment analysis

After defining the most basic concepts and applications related to the field of opinion mining we will have to deal with another important part of it, the different levels of analysis. Sentiment analysis research has been mainly carried out at three levels of granularity : document level, sentence level, and aspect level. We briefly introduce them here.

Document level

At the document level analysis we assume that the overall document-text expresses a unique opinion. Our goal is to classify this opinion as positive, negative or neutral. "It is thus known as document-level sentiment classification" according to [**Liu**, **2015**]. The most notable example is the application of OM to product or movie reviews, one of the first problems studied [**Pang et al.**, **2002 ; Turney**, **2002**], where the goal is to find the sentiment polarity of review, that is, if the review is positive (thumbs-up) or negative (thumbs-down).

It is obvious that such an analysis has many limitations as important information is lost. In a text, many opinions are expressed. For example in a review of a movie, the critic may think that the performances of the actors were good, but the film's direction and scenario are disappointing, resulting in a poor rating of the film. However, this does not reclaim positive opinions about certain aspects of the film, nor does it discern the reasons why the film ended up having a negative rating.

On the other hand, this simple analysis is much easier to implement as there are less properties to be modeled. Observing the definition of opinion given above (2.1.1), in this type of analysis, the opinion holder is one (the author), the object of opinion is one (the film in question) and we do not distinguish its various aspects(of the movie).

Sentence level

At this level of analysis, the goal is equivalent to that of document-level analysis. The only difference is that here we examine the sentiment that expressed in a sentence and not to the whole document. As before, we make a simplistic assumption, that each sentence expresses one (at most) opinion, about one entity. But unlike before we have a deeper level of analysis, which offers the opportunity of mining much more information. This level of analysis is closely related to subjectivity classification, as [**Wiebe et al.**, **1999**] has referred, which distinguishes sentences that express factual information (called objective sentences) from sentences that epress subjective views and opinions (called subjective sentences).

The reason why subjectivity of classification associated with OM is that a non-neutral sentence is by definition subjective. Thus, as a first step when executing OM documents at the sentence-level, the subjective sentences are recognized and then OM procedures are applied only to them.

Aspect level

This level of analysis is the most interesting, but also the most difficult. In the two previous levels of analysis, we make some assumptions, such as how many opinions are contained in a text, in order to simplify the problem. However, in the aspect level analysis in a text, the goal is to identify the sentiment for all the entities and / or their aspects.

The first step is to identify the entities. Then, depending on the degree of analysis, the aspects are extracted for each entity. This is an even more difficult problem, as sometimes one aspect can be indirectly mentioned in the text. In addition, another problem is that basic aspects are referred to in a similar way and therefore it is difficult to recognize that they are the same aspect. These are some significant problems that we'll try to solve, with methods are referred to, in the next paragraphs.

2.2 Preprocessing

An e-text is represented as a combination of characters. Before the processing of the text to extract aspects, usually is required a preprocessing of the text. To preprocess your text simply means to bring your text into a form that is predictable and analyzable for your task. This process, called text preprocessing, consists of different steps, some of which are prerequisites for executing the rest. Thus, depending on the type of analysis we want to apply and the aspects we intend to export, one or more of these steps are performed. "The importance of preprocessing is emphasized by the fact that the quantity of training data grows exponentially with the dimension of the input space" as [**Srivindhya and Anitha**, **2010**] have mentioned. It has already been proven that the time spent on preprocessing can take from 50% up to 80% of the entire classification process [**Morik and Scholz** , **2004**], which clearly proves the importance of preprocessing in the text classification process. We will analyze the most basic preprocessing techniques below.

Tokenization

In this process, a text is transformed from a sequence of characters into a sequence of tokens, such as words, punctuation marks, numbers, etc. This step is necessary to execute the rest as they act on the terms of the text. In English texts, this process is simpler as the words are separated into spaces. However, simply splitting words into spaces is not always enough. For example, "rock 'n' roll ", although consisting of many words that are separated by a space or some punctuation, refers to a specific meaning and would, therefore, be preferred to be kept as a term. In contrast, "I'm" and "doesn't", although not separated by a space, are consisted of separate words-terms ("I am" and "does not").

Therefore, even in languages such as English, this step is very important. Another reason for the importance of correct tokenization use is that all subsequent steps are based on it. Thus, any errors will be promoted in the next steps of preprocessing, but also in extracting aspects and generally to the creation and development of an information retrieval system. Such expressions are :

1. **Words separated by dashes**
   Examples : "over-consumption", "anti-american", "mind-blowing", etc.

2. **Emoticons**
   Examples : ":-)", ":-D", etc.

3. **Slangs**
   Examples : "f**k", "s**t", etc.

4. **Emphasis words**
   Examples : "a * great * time", "I don't * think * I know ...", etc.

Normalization

A highly overlooked preprocessing step is text normalization. Text normalization is the process of transforming a text into a canonical (standard) form. For example, the word "gooood" and "gud" can be transformed to "good", its canonical form. Another example is mapping of near identical words such as "stopwords", "stop-words" and "stop words" to just "stopwords".

Text normalization is important for noisy texts such as social media comments, text messages and comments to blog posts where abbreviations, misspellings and use of out-of-vocabulary words (oov) are prevalent. Here's an example of words before and after normalization:

Original words	**Normalized words**
2moro 2mrrw 2morrow 2mrw tomrw	tomorrow
b4	before
otw	on the way
:) :-) ;-)	smile

Table 2.1: Example of words before and after normalization.

StopWord Removal

Stopwords are a set of commonly used words in a language. Examples of stopwords in English are "a", "the", "is", "are" and etc. The intuition behind using stopwords is that, by removing low information words from text, we can focus on the important words instead.

For example, in the context of a search system, if your search query is "what is text preprocessing?", you want the search system to focus on surfacing documents that talk about text preprocessing over documents that talk about what is. This can be done by preventing all words from your stopword list from being analyzed. Stopwords are commonly applied in search systems, text classification applications, topic modeling, topic extraction and others.

Here is an example of stopword removal in action. All the stopwords are replaced with a dummy character, **W** :
original sentence = this is a text full of content and we need to clean it up
sentence with stopwords removed = W W W text full W content W W W W clean W W

Stopword lists can come from pre-established sets or you can create a custom one for your domain. Some libraries (e.g. sklearn) allow you to remove words that appeared in X% of your documents, which can also give you a stopword removal effect. Finally, there are many different stopword removal methods as they referred to [**Vizayarani et al.**, **2015**].

Stemming

Stemming is the process of reducing inflection in words (e.g. troubled, troubles) to their root form (e.g. trouble). The "root" in this case may not be a real root word, but just a canonical form of the original word. "Stemming converts words to their stems, which incorporates a great deal of language-dependent linguistic knowledge. Behind stemming , the hypothesis is that words with the same stem or word root mostly describe same or relatively close concepts in text and so words can be conflated by using stems" according to [**Srividhya and Anitha**, **2010**].

Stemming uses a crude heuristic process that chops off the ends of words in the hope of correctly transforming words into its root form. So the words "trouble", "troubled" and "troubles" might actually be converted to troubl instead of trouble because the ends were just chopped off.

As they referred to [**Vizayarani et al.**, **2015**] there are many different algorithms for stemming. The most common algorithm, which is also known to be empirically effective for English, is Porter's Algorithm. Here is an example of stemming in action with Porter Stemmer:

Stemming is useful for dealing with sparsity issues as well as standardizing vocabulary. However, as we can observe in Table 2.2 there are stemmed words(e.g troubl or studi) that are not regular words.

Original words	**Stemmed words**
connect connected connection connections connects	connect
trouble troubled troubles	troubl
study studies studied studying	studi

Table 2.2: Example of words before and after stemming with Porter Stemmer.

Lemmatization

Lemmatization on the surface is very similar to stemming, where the goal is to remove inflections and map a word to its root form. The only difference is that, lemmatization tries to do it the proper way. It doesn't just chop things off, it actually transforms words to the actual root. For example, the word "better" would map to "good". It may use a dictionary such as WordNet for mappings or some special rule-based approaches. Here is an example of lemmatization in action using a WordNet-based approach:

Original words	**Lemmatized words**
was were is are	be
trouble troubled troubles troubling	trouble
car cars car's cars'	car
goose geese	goose

Table 2.3: Example of words before and after Lemmatization with WordNet.

Finally, before lemmatization can be executed it is necessary to have the parts of the speech identified , so that in order to find the right lemma it must be known what part of the speech it is.

Noise Removal

Noise removal is one of the most essential text preprocessing steps. It is about removing characters digits and pieces of text that can interfere with your text analysis and it is also highly domain dependent.

There are various ways to remove noise. This includes punctuation removal, special character removal, numbers removal, html formatting removal, domain specific keyword removal (e.g. 'RT' for retweet), source code removal, header removal and more. It all depends on which domain you are working in and what entails noise for your task. Here is an example of noise removal:

Original words	**Cleaned words**
...trouble...	trouble
#trouble	trouble
trouble!	trouble
1.trouble	trouble

Table 2.4: Example of words before and after Remove Noise.

Negation Handling

The sentiment polarity of a word is usually reversed when it is in the context of a negation. " When a negation appears in a sentence it is important to determine the sequence of words which are affected by this term. The scope of negation may be limited only to the next word after a negation or may be extended up to other words following negation" according to [**Farooq et al.**, **2016**]. All the negation words are categorized in three classes, i.e. syntactic, diminisher and morphological negations and they are represented to the Table 2.5 below;

Negation Class	**Negations**
Syntactic	no, not, rather, couldn't, wasn't, didn't, wouldn't, shouldn't, weren't, don't, doesn't, haven't, hasn't, won't, wont, hadn't, never, none, nobody, nothing, neither, nor, nowhere, isn't, can't, cannot, mustn't, mightn't, shan't, without, needn't,
Diminisher	hardly, less, little, rarely, scarcely, seldom
Morphological	Prefixes: de-, dis-, il-, im-, in-, ir-, mis-, non-, un- , Suffix: -less

Table 2.5: List of Negations.

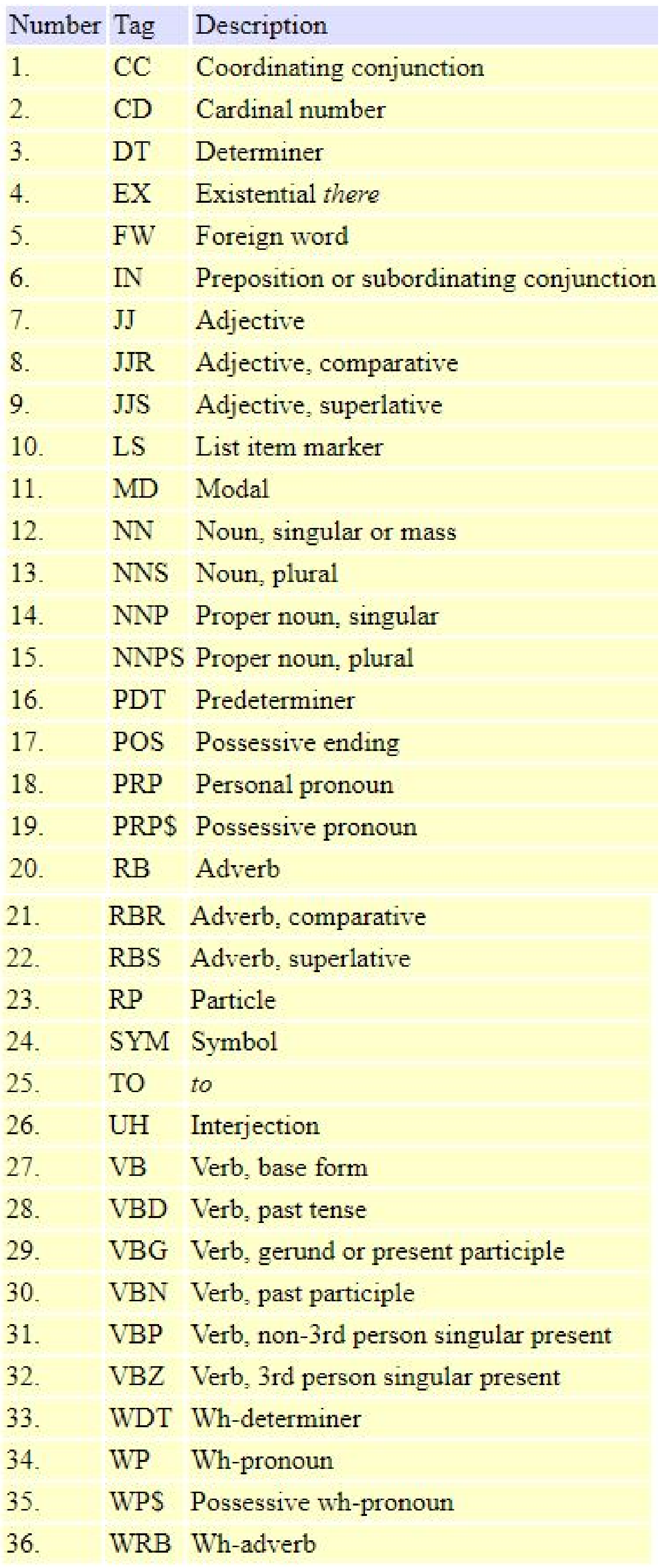
To this end, a common practice is to add the suffix_NEG to any word that is in the context of the negation.
For example the sentence : **No one enjoyed this movie.**
is conversed to : **No one_NEG enjoyed_NEG this_NEG movie_NEG.**

Part-of-Speech Tagging

The Part-of-Speech (POS) Tagging process corresponds to each of the terms extracted from tokenization to a single part of speech from a predefined list (verb, adjective, adverb, etc.) and you can use it to get more granular information about the words in your text. The list mentioned above can be very general, with only a few parts of speech, or quite detailed, distinguishing between different types of adverbs or verbs and other parts of speech. In Table 2.6, on the next page, there is an alphabetical list of part-of-speech tags used in the Penn Treebank Project.

Identifying part of speech tags is much more complicated than simply mapping words to their part of speech tags. This is because POS tagging is not something that is generic. It is quite possible for a single word to have a different part of speech tag in different sentences based on different contexts. That is why it is impossible to have a generic mapping for POS tags.

It is not possible to manually find out different part-of-speech tags for a given corpus. New types of contexts and new words keep coming up in dictionaries in various languages, and manual POS tagging is not scalable in itself. That is why we rely on machine-based POS tagging.



Number	Tag	Description
1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PRP	Personal pronoun
19.	PRP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative
22.	RBS	Adverb, superlative
23.	RP	Particle
24.	SYM	Symbol
25.	TO	*to*
26.	UH	Interjection
27.	VB	Verb, base form
28.	VBD	Verb, past tense
29.	VBG	Verb, gerund or present participle
30.	VBN	Verb, past participle
31.	VBP	Verb, non-3rd person singular present
32.	VBZ	Verb, 3rd person singular present
33.	WDT	Wh-determiner
34.	WP	Wh-pronoun
35.	WP$	Possessive wh-pronoun
36.	WRB	Wh-adverb

Table 2.6: Alphabetical list of POS tags used in Penn Treebank Project. [**Anon, 2019**]

2.3 Thesauri

Most logophiles consider the thesaurus to be a treasure trove of diction, but the word *thesaurus* really does mean "treasure". It derives from the Greek word *thesauros*, which means a storehouse of precious items, or a treasure.

Definition 2.3.1 (Thesaurus) : A thesaurus (pl. thesauruses or thesauri) is a set of items (phrases or words) plus a set of relations between these items as [**Jing and Croft**, **1994**] defined it. It groups words or phrases according to similarity (synonyms, antonyms, hypernyms etc.).

In the early 1530s, a French printer named Robert Estienne published *Thesaurus Linguae Latinae*, a comprehensive Latin dictionary listing words that appeared in Latin texts throughout an enormous span of history. Later, in 1572, Estienne's son Henri published *Thesaurus Linguae Graecae*, a dictionary of Greek words. Although the Estiennes's books were called thesauruses, they were really dictionaries comprised of alphabetical listings of words with definitions, as they referred to [**Mentafloss.com**, **2019**]. This is a good example for clarifying the difference between dictionaries and thesauruses.

Specifically, a dictionary is a collection of words along with their meaning, definition and description of usage while a thesaurus presents words as "word families" listing their synonyms without explaining their meanings or usage. In addition, thesauri may list words alphabetically or conceptually, but dictionaries always list words alphabetically.

Thus, the first modern thesaurus was written by a British doctor, Peter Mark Roget. In 1805, he began compiling a list of words, arranged by their meaning and grouped according to theme. After retiring from his work as a physician in 1852, Roget published *Thesaurus of English words and phrases; so classified and arranged as to facilitate the expression of ideas and assist in literary composition.* Roget's Thesaurus, a collection of words and phrases arranged according to the ideas they express, presents a solid framework for a lexical knowledge base. Its explicit ontology offers a classification system for all concepts that can be expressed by English words; the original system was organized in seven classes: 1) Abstract relations, 2) Space, 3) Material World, 4) Intellect, 5) Volition, 6) Sentient and 7) Moral Powers; it is beneficial for NLP experiments and there are many versions of it in [**Mario Jarmasz**, **2003**]. Today, Roget's Thesaurus is still commercially successful and widely used. In fact, we celebrate Thesaurus Day on January 18 because Roget was born on this day in 1779.

WordNet (George Miller,1960) is also such an instrument and it has proven to be invaluable to the NLP community. "Although it is an electronic lexical database based on psycholinguistic principles, it has been used almost exclusively in NLP" according to [**Mario Jarmasz**, **2003**]. The semantic relations in WordNet have often been studied, even exploited for a variety of applications, for example measuring semantic similarity [**Budannitsky and Hirst**, **2001**], and encoding models for answers types in open-domain question answering systems [**Pasca and Harabagiu**, **2001**]. Using WordNet as a blueprint, various multilingual lexical databases have been implemented, the first one being EuroWordNet and it is presented at [**Vossen**, **1998**]. It is a multilingual electronic lexical database for Dutch, Italian, Spanish, German, French and Estonian.

As a result, because of the utility of WordNet in many applications and the limitations of the printed version of Roget's Thesaurus, many researchers have opted for WordNet when attempting to extend their algorithms beyond their problems.

Beyond the definition of the thesaurus, its differences with the dictionary and historical pieces of information about the most popular thesauruses (Roget's and WordNet), two are the major thesauruses' issues that we analyze below:

2.3.1 Types of thesauri

There are two types of thesauri, manual and automatic. Thesauruses such as Roget and WordNet are produced manually, whereas others, as in pioneering work by [**Karen**, **1986**] and more recent advances from [**Grefenstette**, **1994**] and [**Lin**, **1998**] are produced automatically from text corpora. One might consider the manually-produced ones to be semantic, since lexicographers put words in the same group according to their meaning, whereas the automatically produced ones are distributional since the computer classifies them according to distribution. However, there are many differences in viewing them as different sorts of objects.

Manual thesauri are evaluated in terms of the soundness, coverage of classification and thesaurus item selection while the evaluation of automatic thesauri is usually done via query-expansion to see if retrieval performance is improved [**Jing and Croft**, **1994**]. There are two types of manual thesauri : 1) general-purpose and word-based thesauri and 2) IR-oriented and phrase-based thesauri. The first contains sense relations like antonym and synonym but are rarely used in Information Retrieval (IR) (e.g Roget's and WordNet). The second usually contains relations between thesaurus items such as BT (Broader Term), NT (Narrow Term), UF (Used For), and RT (Related To), and can be either general or specific, depending on the needs of thesaurus builders. This type of manual thesauri is widely used in commercial systems and some of the most known examples of them are INSPEC, LCSH (Library of Congress Subject Headings), and MeSH (Medical Subject Headings).

On the other hand, an automatic thesaurus is usually collection dependent, i.e, dependent on the text database which is used. Automatic thesauri are typically built based on co-occurrence information, and relevance judgements are often used to estimate the propability that thesaurus terms are similar to query terms or a particular query.

Finally, the major disadvantage of manual thesauri is that they are expensive to build and hard to update in a timely manner. Even though the determination of thesaurus item relations is made by human experts, it is a difficult task. Automatic thesauri also have a major problem, as a few small and automatically constructed theusari have been used in experimental IR systems but the effectiveness of these thesauri has not been established for large text databases. In conclusion, both types of thesauri are semantic and useful, with both advantages and disadvantages. Thus, in each case, we choose which is better for our research.

2.3.2 Semantic uses of thesauri

There are many applications associated with the use of high-quality thesauruses. The most important are described below:

Parsing

A thesaurus contains salient information for many parsing tasks including the very hard ones (for English and probably other languages) of 1) prepositional phrase (PP) attachment and 2) conjunction scope .

PP-attachment

According to [**Kummerfeld et al.**, **2012**] the Prepositional Phrase (PP) attachment problem is a classic ambiguity problem and is one of the main sources of errors for syntactic parsers.

[**Ratnaparkhi et al.**, **1994**] first proposed a formulation of PP attachment as a binary prediction problem. The task is as follows: we are given a fourway tuple (v, o, p, m) where v is a verb, o is a noun object, p is a preposition, and m is a modifier noun; the goal is to decide whether the prepositional phrase (p, m) attaches to the verb v or to the noun object o.

More recently, [**Belinkov et al.**, **2014**] proposed a generalization of PP attachment that considers multiple attachment candidates. Formally, we are given a tuple (H, p, m), where H is a set of candidate attachment tokens, and the goal is to decide what is the correct attachment for the (p, m) prepositional phrase. The binary case corresponds to H = (v, o).

Below we will explain an example mentioned in [**Madhyastha et al.**, **2017**]. We have two sentences :
Sentence 1 : I went to the restaurant by the Hudson.
Sentence 2 : I went to the restaurant by bike.
For the first sentence, the correct attachment is the prepositional phrase attaching to the *restaurant*, the noun. Whereas, in the second sentence the attachment site is the verb *went*. While the attachments are ambiguous, the ambiguity is more severe when unseen or infrequent words like *Hudson* are encountered.

Conjuction scope

We will enclose this thesaurus' application with a very useful and easy example. We have again two sentences :
Sentence 1 : Old books and newspapers.
Sentence 2 : Old books and clothes.
It cannot be determined with confidence without more context whether the *newspapers* are old, and whether the *clothes* are old. However, one fact suggesting that the *newspapers* are old while the *clothes* are not is that *book* and *newspaper* are close in the thesaurus, and thesaurally close items are frequently found in conjunction, so *books* and *newspaper* is a likely syntactic unit.

Bridging Anaphor Resolution

Definition 2.3.2.1 (Anaphora) : Anaphora is the use of a word referring back to a word used earlier in a text or conversation, to avoid repetition, for example the pronouns he, she, it, and they and the verb do in I like it and so do they.

Anaphora plays a major role in discourse comprehension and accounts for the coherence of a text. There are two main types of anaphor : **1)** identity anaphora and **2)** bridging anaphora. The first one, indicates that a noun phrase refers back to the same entity introduced by previous descriptions in the discourse. In contrast to identity anaphora, bridging anaphora or associative anaphora is based on the nonidentical associated antecedent and links anaphors and antecedents via lexico-semantic, frame or encyclopedic relations [**Hou**, **2018**]. Full bridging resolution combines two subtasks: (i) detecting bridging anaphors (anaphor recognition) and (ii) finding an antecedent for given bridging anaphors (anaphor resolution) [**Rosiger**, **2018**].

We refer the example of [**Kilgarrif**, **2003**] below in order to understand the usefulness of bridging anaphor resolution. On the sentence : (Maria bought a beautiful apple. The fruit was red and crispy.) the fruit and the apple co-refer. The proximity of *fruit* and *apple* in a thesaurus can be used to help an algorithm establish that the *fruit* is a bridging anaphor referring back to *apple*.

Word Sense Disambiguation

Most words in natural languages are polysemous, that is they have multiple possible meanings or senses. For example, as it is referred in [**Kilgarrif**, **2003**] the noun *pike* can mean either a fish and a weapon and is not a common word. In the sentence : (We caught a pike that afternoon.), a human reader immediately knows from the surrounding context that pike refers to a fish because of the verb caught (the past version of catch), and not to a weapon. However, computer programs do not have the benefit of a human's vast experience of the world and language, so automatically determining the correct sense of a polysemous word is a difficult problem. According to [**Banerjee and Pedersen**, **2002**], this process is called word sense disambiguation, and has long been recognized as a significant component in language processing applications such as information retrieval, machine translation, speech recognition, etc.

As it was referred, *pike* is not a common word so there is probably no evidence at our disposal for a direct connection between *catch* and *pike*. However. there is likely to be some evidence connecting catch to one or more word which is thesaurally close to pike such as *roach, bream, carp, cod, mackerel, shark or fish.* Given a thesaurus, we can infer that the meaning of *pike* in this sentence is the fishy one.

Lexical Cohesion

Definition 2.3.2.2 (Lexical Cohesion) : Lexical cohesion refers to the ties created between lexical elements, such as words (e.g nouns, verbs, adjectives, adverbs, etc.) and phrases. These lexical ties can occur over long passages of text or discourse. The primary types of lexical cohesion are : 1) reiteration and 2) collocation.

Reiteration

Reiteration is a form of Lexical cohesion which includes the repetition of a lexical item, synonyms-antonyms, hypernyms-hyponyms, meronyms etc. We analyze the most considerable of them below:

- **Repetition**
  When a lexical item (nouns, verbs, adjectives, adverbs) and variation of the word are repeated, it contributes to the text's overall cohesion.

  Example : *Julia Costello is facing a* ***difficult*** *situation at WesternTechnologies Corporation. She has* ***difficulty*** *functioning in the executive team.*
  There is a repetition chain between difficult and difficulty.

- **Synonyms-Antonyms**
  Lexical cohesion occurs when a word is in some ways **synonymous** (the same) as a word that went before it in the text.

  Example : *He was just wondering which road to take when he was started by a* ***noise*** *from behind him. It was the noise of trotting* <u>*horses*</u>*... He dismounted and led his horse as quickly as he could along the right-hand road. The* ***sound*** *of the* <u>*cavalry*</u> *grew rapidly nearer...*
  We observe that horses-cavarly and noise-sound are two different pairs of synonyms in this example.

  A relational **antonym** is one of a pair of words with opposite meanings, where opposite makes sense only in the context of the relationship between the two meanings.

  Example : *I have a dream that one day, down in Alabama, with its vicious racists, with its governor having his lips dripping with the words of interposition and nullification, one day right there in Alabama, little* ***black*** *boys and girls will be able to join hands with little* ***white*** *boys and girls as sisters and brothers.*

  The words black and white are antonyms.

- **Hypernyms-Hyponyms**
  In linguistics, a **hyponym** (from Greek hupó, "under" and ónoma, "name") is a word or phrase whose semantic field is included within that of another word, its **hyperonym or hypernym** (from Greek hupér, "over" and ónoma, "name"). In simpler terms, a hyponym is in a type-of relationship with its hypernym.

  Example : *pigeon, crow, eagle and seagull are all hyponyms of bird (their hypernym); which, in turn, is a hyponym of animal*

Collocation

This is a type of ‘expectancy relation’ between lexical items. In other words, when a certain word occurs, you expect another word, or words to occur with it. For example, the word **ailment** is likely to be found together with **prescribe, diagnose** and **treat**. In other words, there is a tendency for the items to occur together.

2.4 Term Frequency-Inverse Document Frequency

If they give us a sentence for example “This dress is so beautiful and it is made of high-quality fabric”, it’s easy for us to understand the sentence as we know the semantics of the words and the sentence. But how will the computer understand this sentence ?

The computer can understand any data only in the form of numerical value. So for this reason we vectorize all of the text so that the computer can understand the text better. In addition, by vectorizing the documents we can perform multiple tasks such as finding the relevant documents, ranking, clustering and so on. Several approaches have been developed for converting text to numbers. 1) Bag of Words, 2) N-grams and 3) Word2Vec model are some of them. We will describe the Bag-of-Words (BoW) model below, because Term Frequency-Inverse Document Frequency (TF-IDF) is related to this.

In the BoW approach, the vocabulary of all the unique words in all the documents-sentences is formed. This vocabulary serves as a feature vector. Suppose you have three documents in our corpus :

$$S_1 = \text{“It is cold outside.”}$$

$$S_2 = \text{“The weather is cold.”}$$

$$S_3 = \text{“I am outside.”}$$

The vocabulary formed using the above three sentences will be :
V=[it, is, cold, outside, the, weather, I, am] .
This vocabulary of words will be used to create feature vectors from the sentence. Basically, the feature vector is created by finding if the word in the vocabulary is also found in the sentence. If a word is found in vocabulary as well as in the sentence, a one (1) is entered in that place, else a zero(0) will be entered. Thus, the feature vectors for the previous sentences will be:

$$S_1 = [1, 1, 1, 1, 0, 0, 0, 0]$$

$$S_2 = [0, 1, 1, 0, 1, 1, 0, 0]$$

$$S_3 = [0, 0, 0, 1, 0, 0, 1, 1]$$

In a simple Bag-of-Words, every word is given equal importance. The idea behind TF-IDF is that the words that occur more frequently in one document and less frequently in other documents should be given more importance as they are more useful for classification. Thus, in order to explain the relationship between BoW and TF-IDF and the utility of the second, we refer to the definition and the whole method of measurement of this semantic statistical measure.

Definition 2.4.1 (TF-IDF) : TF-IDF which stands for Term Frequency-Inverse Document Frequency , is a statistic widely used in IR and summarization and measures how important a term is relative to a document and to a corpus.

Terminology : 1) ***t***-term, 2) ***d***-document(set of words), 3) ***N***-count of corpus (corpus is the total document set)

Now we will analyze the two parts of TF-IDF :

Term Frequency

Term Frequency (TF) gives the frequency of a word in each document in the corpus. It is the ratio of number of times the word appears in a document compared to the total number of words in that document.It increases as the number of occurences of that word within the document increases.

In the previous example the TF in S_1 for the word "outside" will be:

$$\frac{1}{4} = 0.25$$

TF depends on the length of the document and the generality of the word. For example, a very common word such as "was" can appear multiple times in a document. But if we take two documents one which have 100 words and other which have 10.000 words, there is a high propability that the common word such as "was" can be present more in the 10.000 worded document. However, we cannot say that the longer document is more important than the shorter doc. For this exact, we perform a normalization on the frequency value. We divide the frequency with the total number of words in the document. Tf is individual to each document and word, hence we can formulate it as follows :

$$tf(t,d) = \frac{\text{count of t in d}}{\text{number of words in d}} \quad \textbf{(2.1)}$$

If we already computed the TF value and if this produces a vectorized form of the document , why not use just TF to find the relevance between documents and we need IDF?

Inverse Document Frequency
The answer is coming from the Document Frequency. This measures the importance of document in whole set of corpus for a term *t* in document d, where as DF is the count of occurences of term *t* in the document set *N*. In other words, DF is the number of documents in which the word is present. We consider one occurence if the term consists in the document at least once, we do not need to know the number of times the term is present. We refer the type for DF below:

$$df(t) = \text{ occurence of t in documents } \textbf{(2.2)}$$

To keep this also in a range, we normalize by dividing with the total number of documents. The main goal is to know the informativeness of a term, and DF is the exact inverse of it; that is why we inverse DF.

Inverse Document Frequency (IDF) is the inverse of the document frequency and is used to calculate the weight of rare words across all documents in the corpus. The words that occur rarely in the corpus have a high IDF score (Figure 2.1).

The type of IDF is :

$$idf(t) = \frac{N}{df} \ \textbf{(2.3)}$$

Now there are few other problems with the IDF, in case of a large corpus, say 10.000, the IDF value explodes. So to dampen the effect we take *log* of IDF. Thus, the final type of IDF is :

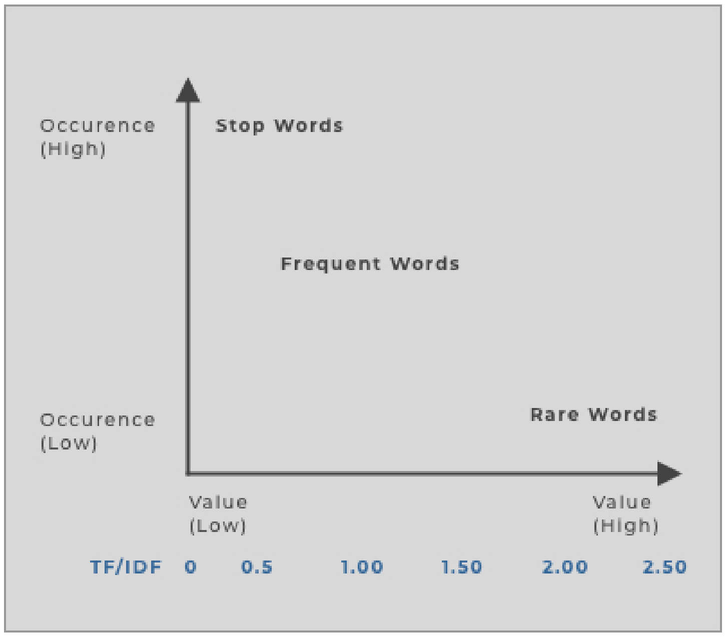


Figure 2.1: Relation between the ocurrence of a word and TF-IDF. [**Boecher**, **2019**]

$$idf(t) = \log \frac{N}{df} \ \textbf{(2.4)}$$

In the previous example the IDF for the word “ outside “ in S_1 will be:

$$\log \frac{3}{2} = 0.176$$

Finally, by taking multiplicative value of TF (2.1) and IDF (2.4), we get the basic version of TF-IDF:

$$tf - idf(t) = tf(t, d) * idf(t) \ \textbf{(2.5)}$$

Thus, by applying (2.5) in the previous example we get tf-idf= 0.25 * 0.176 = 0.044 .

Chapter 3

Methodological Considerations

As it is mentioned in paragraph 1.2 of Chapter 1, the purpose of this dissertation is the design and development of an intelligent information retrieval system. In order to create and develop this system, three semantic processes are required. The first two, are related to the pre-processing of the data and user's query and to the extraction of aspect-words and are described in paragraphs 3.1 and 3.2, respectively. The third procedure is referred to as the connection of the user's query with the corresponding data, and the retrieving of the most relevant document(s) that express opinion. The third, as well as the whole system, is described in more detail in the third and final paragraph of this chapter.Before we describe each method individually we summarized the total system and its methods on the next page to the Figure 3.1. Finally, the implementation of the algorithm for the construction of the system mentioned above has been in Python programming language.

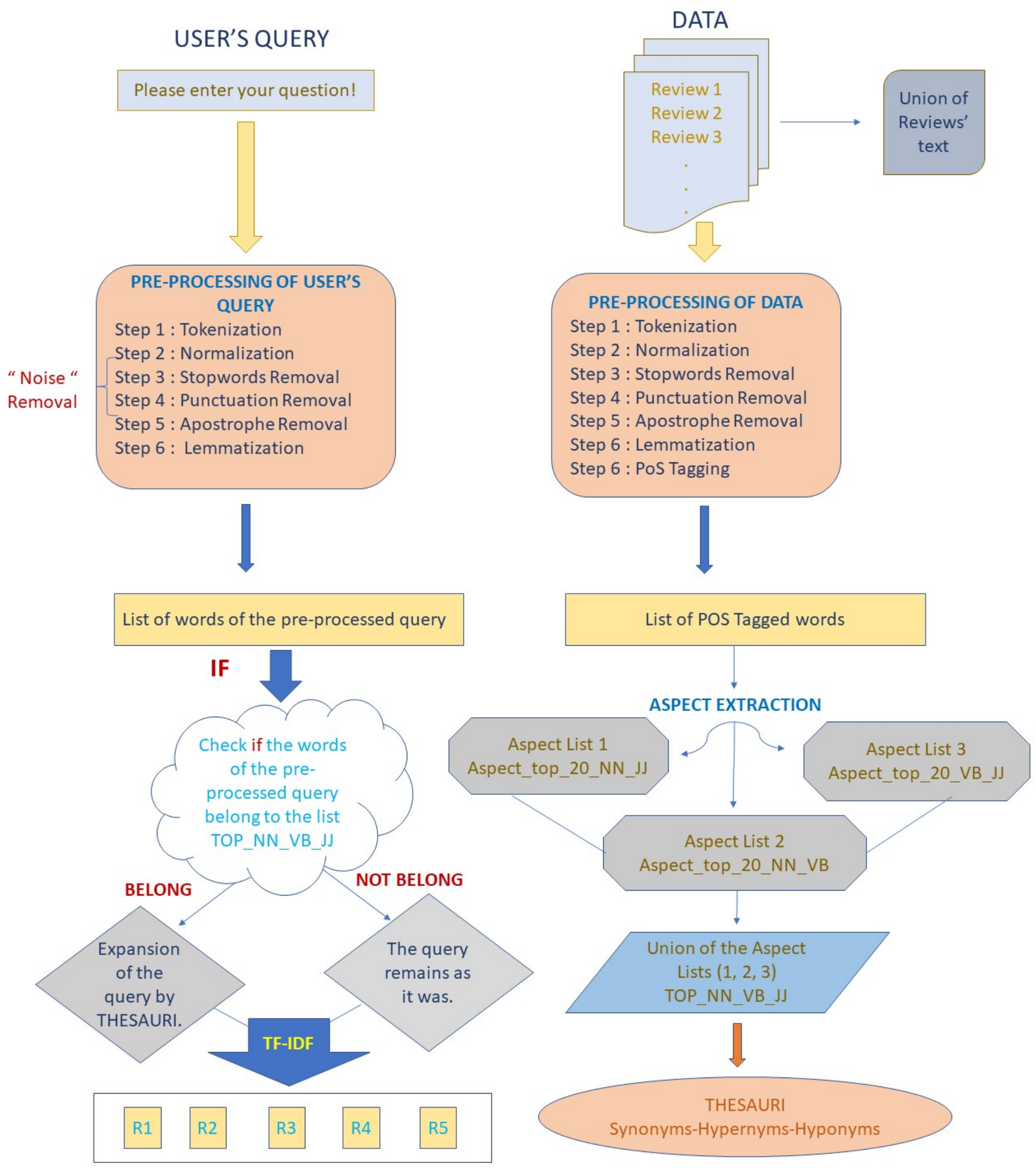

USER'S QUERY
Please enter your question!
PRE-PROCESSING OF USER'S QUERY
Step 1 : Tokenization
Step 2 : Normalization
Step 3 : Stopwords Removal
Step 4 : Punctuation Removal
Step 5 : Apostrophe Removal
Step 6 : Lemmatization
" Noise " Removal
List of words of the pre-processed query
IF
Check if the words of the pre-processed query belong to the list TOP_NN_VB_JJ
BELONG
NOT BELONG
Expansion of the query by THESAURI.
The query remains as it was.
TF-IDF
R1
R2
R3
R4
R5
DATA
Review 1
Review 2
Review 3
Union of Reviews' text
PRE-PROCESSING OF DATA
Step 1 : Tokenization
Step 2 : Normalization
Step 3 : Stopwords Removal
Step 4 : Punctuation Removal
Step 5 : Apostrophe Removal
Step 6 : Lemmatization
Step 6 : PoS Tagging
List of POS Tagged words
ASPECT EXTRACTION
Aspect List 1 Aspect_top_20_NN_JJ
Aspect List 3 Aspect_top_20_VB_JJ
Aspect List 2 Aspect_top_20_NN_VB
Union of the Aspect Lists (1, 2, 3) TOP_NN_VB_JJ
THESAURI Synonyms-Hypernyms-Hyponyms

Figure 3.1: Design of the Information Retrieval System.

3.1 Pre-processing of data and user's query

Data pre-processing was essential, as many of the datasets that have been used on this research have been on the scale of millions or even billions. This also saved a lot of time and resources, which made the system quite fast, efficient and flexible in many different data domains.

In addition to data pre-processing, it was also necessary this of user's query. The purpose was the encoding of the query and the formation of it in the latter of keywords, so that it can be easily understood by the system. This could only be done by filtering and eliminating all the useless information, as it did. Therefore, this also helped in saving time and resources and enhanced the accuracy of the system.

Pre-processing of data and user's query was, therefore, the first part of the system's creation and it is described below;

Data Preprocessing
Data that have been used for the algorithm were products' reviews from Amazon. Every review consisted of many features (more information in paragraph 4.1), but only the text of reviews has been used for the reviews' preprocessing. Moreover, Natural Language Toolkit *(NLTK)*, which is an open-source Python library for Natural Language Processing (NLP), and *NumPy* package, were necessary for the implementation of the seven steps of data's preprocessing. The seven steps are described below.

1. **Tokenization**

   First of all, the text of each review was transformed into tokens. Specifically, by importing *sent_tokenize* and *word_tokenize* from the *NLTK* library, the review text was transformed from a sequence of characters to chunks of sentences and words, respectively. However, only the texts that have been tokenized into words and the union of them as a united word-list, have been used in the next steps.

2. **Normalization**

   Programming languages consider textual data as sensitive, which means that "The" is different from "the". We, humans, know that both those belong to the same token but due to the character encoding, those are considered as different tokens. This problem has been resolved in this step by converting to lowercase. As we had all our data in a word-list, *NumPy's* method (*np.char.lower(data)*) has been used and converted our list in lowercase at once.

3. **Stopwords Removal**

   Stopwords are the most commonly occurring words which don't give any additional value to the document vector. Thus, in step three these words have been removed by two ways. The first and the most simple way was to download a stopword list from *NLTK* library and by iterating to our word-list remove all the stopwords. The second, was to create our stopword-list from scratch, with words that have been observed to be repeated and they didn't give us any useful information. Thus, by removing these words computation and space efficiency have been increased.

4. **Punctuation Removal**

   The punctuation marks are a form of " noise " as it was mentioned in section 2.2. As a large percentage of the tokens are punctuation symbols, it was necessary to remove them from the text, as we did on step four. All the tokens, that have been also punctuation marks of the *string.punctuation* word-list of *NLTK*, have been removed from our word-list.

5. **Apostrophe Removal**

   During the punctuation removal, you can note that apostrophe ' is not contained in punctuation list *string.punctuation*=[! " # $ % & ' ( ) * + , - . / : ; ¡ = ¿ ? @ [ ] ˆ_ { — }]. Also, after punctuation removal, you can observe that there are some words which are stopwords and they won't be removed (e.g don't that converted to dont). Thus, except of stopwords and punctuation we had to remove the apostrophe of these words. This was accomplished with *NumPy's* method (*np.char.replace (data, " ' ", " ")*).

6. **Stemming-Lemmatization**

   Firstly, Porter-Stemmer, which is a rule-based stemmer and removes the suffix or affix of a word, has been used in step six. Nevertheless, many words of our word-list changed and lost their meaning (e.g love→lov, wine→win, etc.). Hence, stemmed words have not been used in our system. On the contrary, we lemmatized the words of our word-list with *WordNetLemmatizer*. The result of the lemmatization procedure was quite satisfactory, as it reduced words to root synonym words, and a lot of words with the same meaning transformed to the same word (e.g dish, dishes→dish).

7. **POS -Tagging**

   After tokenization, normalization, lemmatization, and removal of "noise" such as stopwords, apostrophe and punctuation symbols, the last step was the part-of-speech tagging of the chunked words. So, the method (*nltk.pos_tag(data)*) of *NLTK* library, inserted tags-labels in each word of our word-list and we got information about their kind(e.g noun, adjective, adverb, and etc.). This was very useful and has been used in the next part of the system's design that we will analyze in the next paragraph(3.2).

Preprocessing of user's query
Users must be able to express their information need in both, an exact and a fuzzy way. In other words, we want to offer the possibility to express various kinds of queries, ranging from "approximate" ones, where no knowledge about the structure and the characteristics of the underlying data sources is available, to more exact ones, where users know the underlying data sources and their query capabilities as well as the (kind of) information they can expect.

However, this "freedom" of the Information Retrieval System, where "everyone can ask everything", sometimes causes problems, such as delays and complicated queries in which the needs of the user are not clearly expressed. Thus, pre-processing of user's query was of great importance for the system's improvement.

This includes the same steps as data's preprocessing, except for the last step of Part-of-Speech tagging. More specifically, the queries' text was tokenized into words and these tokens have been saved into a word-list. Then, normalization of them was the second step of the procedure, while the next three steps consisted of the "noise" removal part. As "noise" could be stopwords, punctuation marks or apostrophe symbol. Finally, the last part of query's pre-processing was that of lemmatization. In this way, the queries have been transformed in the latter of keywords.

After the first process of the system's design has been completed, data was ready for the aspect words' extraction and for the research of how relative the question is to them. These processes, while others procedures of the system's creation, such as the extension of the queries with synonyms, hypernyms or hyponyms of thesauruses, will be described in the next sections.

3.2 Extraction of aspect words

Researchers have studied opinion mining at the document, sentence and aspect levels as we referred in section 2.1.3. Aspect-level (called aspect-based opinion mining) is often desired in practical applications as it provides detailed opinions or sentiments about different aspects of entities and entities themselves, which are usually required for action. Aspect extraction and entity extraction are thus two core tasks of aspect-based opinion mining. In this section, we refer to these tasks and to the current state-of-the-art extraction techniques. Finally, we describe in detail the method of aspects' extraction that has been used to our Information Retrieval System.

According to [**Sarawagi**, **2008**] aspect and entity extraction are both fall into the broad class of information extraction whose goal is to automatically extract structured information (e.g., names of persons, organizations and locations, brands of products, etc.) from unstructured sources. Traditional information extraction techniques are often developed for formal genre (e.g., news, scientific papers, etc.), but we aim to extract fine-grained information from opinion documents (e.g., reviews, blogs and forum discussions). These are often very noisy and also have some distinct characteristics that can be exploited for extraction. Hence, the design of extraction methods that are specific to opinion documents are essential. In addition, as aspect extraction and entity extraction are closely related, some techniques that proposed for aspect extraction can be applied to the task of entity extraction as well. Thus, we focus on aspect extraction methods that are referred to [**Zhang and Liu**, **2014**] and we descibe them below;

Exploiting Language Rules

Language rule-based approaches are the oldest in information extraction and they tend to focus on patterns-matching or parsing. These patterns capture various properties of one or more terms and their relations (e.g grammatical relations between aspects and opinion words or other terms) in the text.

The first and most common method of extraction aspects based on asssociation rules proposed by [**Hu and Liou**, **2004a**] and it is consisted of two parts: 1) finding frequent nouns and noun phrases as frequent aspects and 2) using relations between aspects and opinion words to identify infrequent aspects.

In the first step, nouns and noun phrases are identified by a part-of-speech (POS) tagger and only the k (k=5 or 10 or 20 or etc. and it is a threshold) most frequent are kept. These are quite important aspects as the most times when people comment on different aspects of a product, the vocabulary that they use usually converges. In contrast, reviews that contain irrelevant and infrequent nouns, that likely to be non-aspects or less important aspects, are often diverse.

The second one, tries to find many aspect expressions which are infrequent and they missed in the first step.The idea is summarized as: the same opinion word can be used to describe or modify both different frequent and infrequent aspects.

Sequence Models

One more aspect extraction technique is this of Sequence models. They have been widely used in information extraction tasks and can be applied to aspect extraction as well. Because of the interdepedence of product aspects, entities and opinion expressions and that occur at a sequence in a sentence, we can assume that aspect extraction is a sequence labeling task. The most known sequence models are 1) Hidden Markov Model [**Rabiner**, **1989**] and 2) Conditional Random Fields [**Lafferty et al.**, **2001**].

Topic Models

In machine learning and natural language processing, a topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents and it is a frequently used text-mining tool for discovery of hidden semantic structures in a text body.

In other words, a topic model is a generative model for documents that it specifies a probabilistic procedure by which documents can be generated. When someone wants to construct a new document, then he/she is choosing a distribution D over topics and for each word in that document, a topic randomly is chosen according to D and is drawn a word from the topic.

Latent Dirichlet Allocation (LDA) according to [**Blei**, **2003**] is an example of topic model and is used to classify text in a document to a particular topic.

Miscellaneous Methods

[**Yi et al.**, **2003**] proposed a method for aspect extraction based on the likelihood-ratio test. [**Bloom et al.**, **2007**] manually built a taxonomy for aspects, which indicates aspect type. They also constructed an aspect list by starting with a sample of reviews that the list would apply to. They examined the seed list manually and used WordNet to suggest additional terms to add to the list. [**Lu et al.**, **2010**] exploited the online ontology Freebase to obtain aspects to a topic and used them to organize scattered opinions to generate structured opinion summaries. [**Ma and Wan**, **2010**] exploited Centering theory [**Grosz et al.**, **1995**] to extract opinion targets from news comments. The approach uses global information in news articles as well as contextual information in adjacent sentences of comments. Finally, [**Yu et al.**, **2011**] used a partially supervised learning method called one-class SVM to extract aspects.

After describing the most well-known attribute extraction methods it is time for our aspect extraction technique that applied to the data.

As it was mentioned in section 3.1 of this chapter, the last step of data's pre-processing was the POS Tagging of the words that we had stored in a word list after we had applied all the previous steps of pre-processing. In this step, by comparing the tagging of the labeled words and the Table 2.6 (page 21), we have noticed that these words can be various parts of speech such as nouns (NN), verbs (VB), adjectives (JJ), adverbs (RB), preposition or subordinating conjunction (IN), modal (MD), interjection (INT) and etc. However, what is not unnoticed is that most of these words are either nouns, adjectives or verbs, which is not a random fact.

These three parts of speech are essential to a text and constitute the most information it can provide. More specifically :

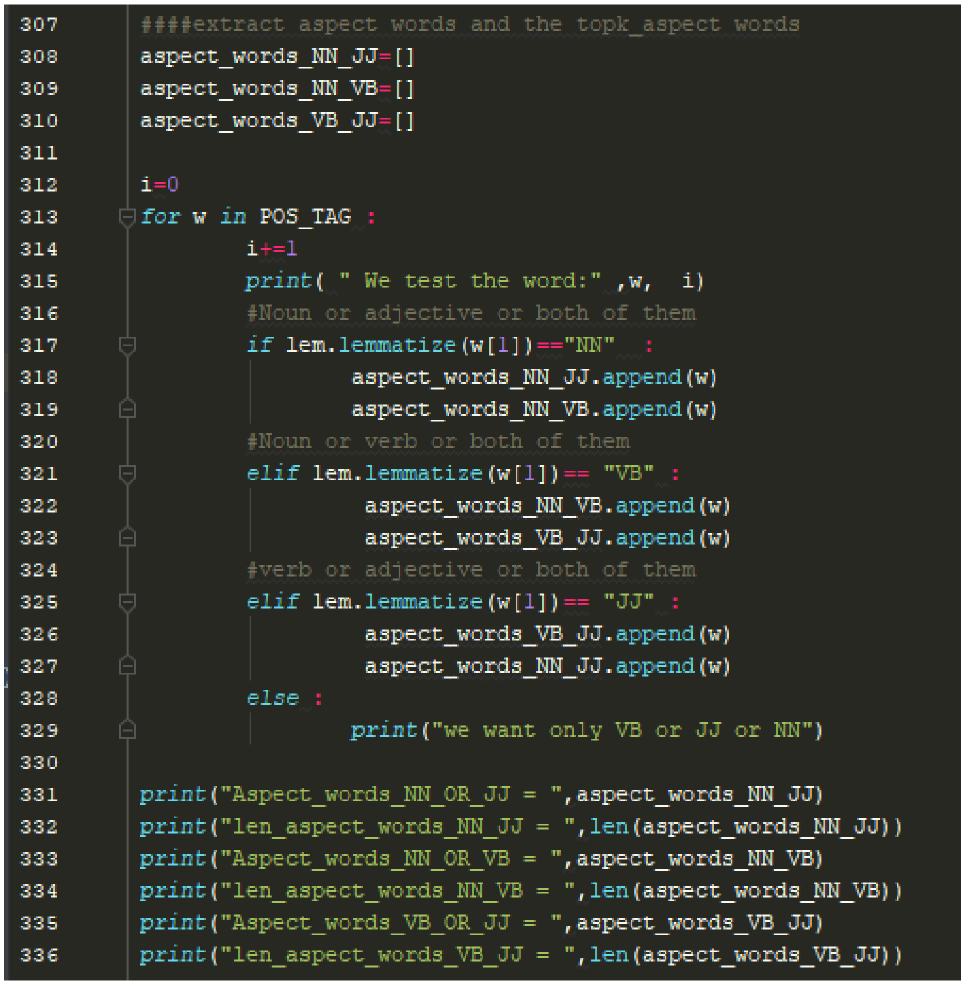
Nouns-NN : Nouns generally refer to people, places, things, or concepts, e.g.: woman, Scotland, book, intelligence. Nouns can appear after determiners and adjectives, and can be the subject or object of the verb.
Verbs-VB : Verbs are words that describe events and actions, e.g. fall, eat. In the context of a sentence, verbs typically express a relation involving the referents of one or more noun phrases.
Adjectives-JJ : Adjectives describe nouns, and can be used as modifiers (e.g. large in the large pizza), or in predicates (e.g. the pizza is large). English adjectives can have internal structure (e.g. fall+ ing in the falling stocks).

Therefore, nouns, verbs and adjectives are directly interdependent in a text.

Usually, there is a core-noun (the most frequent noun) which is the entity we want to extract while the rest of the nouns that may appear in the text as well as adjectives are the aspects of it and they are the most significant part of our extraction. Finally, the verb(s) of the text express the action of the noun(s) that mentioned above.

By regarding all the above details as well as the fact that there are words that can be two parts of speech (e.g *love* is a verb (VB) but it is also a noun (NN), *taste* is a verb (VB) but is also a noun (NN), *cheese* is a noun (NN) but is also an adjective (JJ), and etc.) at the same time, we have kept three lists of tagged words that can only be nouns or verbs or adjectives. We descibe them below and it is also incuded part of the algorithm of their creation (Figure 3.2).

The first list includes all the words that have been tagged as nouns or adjectives on the POS Tagging step. Respectively, the second one contains nouns and verbs and the last is the list of the verbs and adjectives (lines 308-336 in Figure 3.2). Most of the time, the list of most words is the first, while the list of the few is the last one. Then we reduced the size of these three lists by creating new lists which contain only the twenty most frequent nouns-adjectives, nouns-verbs and verbs-adjectives, respectively. The last are the lists that we have expanded with the synonyms, hypernyms and hyponyms and thus we created the thesauri that we will explain in the next and final section.



```
####extract aspect words and the topk_aspect words
aspect_words_NN_JJ=[]
aspect_words_NN_VB=[]
aspect_words_VB_JJ=[]

i=0
for w in POS_TAG :
        i+=1
        print( " We test the word:" ,w,  i)
        #Noun or adjective or both of them
        if lem.lemmatize(w[1])=="NN"  :
                aspect_words_NN_JJ.append(w)
                aspect_words_NN_VB.append(w)
        #Noun or verb or both of them
        elif lem.lemmatize(w[1])== "VB" :
                 aspect_words_NN_VB.append(w)
                 aspect_words_VB_JJ.append(w)
        #verb or adjective or both of them
        elif lem.lemmatize(w[1])== "JJ" :
                 aspect_words_VB_JJ.append(w)
                 aspect_words_NN_JJ.append(w)
        else :
                print("we want only VB or JJ or NN")

print("Aspect_words_NN_OR_JJ = ",aspect_words_NN_JJ)
print("len_aspect_words_NN_JJ = ",len(aspect_words_NN_JJ))
print("Aspect_words_NN_OR_VB = ",aspect_words_NN_VB)
print("len_aspect_words_NN_VB = ",len(aspect_words_NN_VB))
print("Aspect_words_VB_OR_JJ = ",aspect_words_VB_JJ)
print("len_aspect_words_VB_JJ = ",len(aspect_words_VB_JJ))
```

Figure 3.2: Part of the algorithm of the aspect extraction method.

3.3 Representation of the Information Retrieval System

At the beginning of the chapter we mentioned that three processes were needed for the design and development of our information retrieval system. The first two, of data and user query's pre-processing and this of aspects' extraction, were analyzed in the two previous sections, respectively. The final section of chapter 3 contains the last one, which is the connection between user's query and the data. Before explaining this connection, however, it is necessary to explain the other two extra methods that were needed in order to make the system more efficient. These are the following;

Top-aspects' expansion based on Thesauri

We have completed the previous section by referring to the creation of the lists which contain the top 20 aspect words. However, the fact that these three lists contain aspect words that many of them are the same and they constitute the largest percentage of information we can retrieve from our data, has led us to unite them. Thus, we created the union of them, a general list of aspects (lines 366-380 in Figure 3.3).

At this point, however, we have to be wondered if there were several aspect words of the united aspect word list to extract information or there were similar or related words that might also appear in our data, and they could also be important information to us.

The answer was in the thesauruses and the idea of the creation of a united aspect list was completed when it was expanded. This expansion was based on thesauri with synonyms, hypernyms, and hyponyms. Specifically, by importing *WordNet* from *NLTK* library in Python and by using methods of it that are described in the part of algorithm in Figure 3.3 (lines 384-400), we created three lists(*synonyms*, *hypernyms*, *hyponyms*) that constitute three different extentions of each aspect word of the united aspect word list.

For example the aspect word *wine* of :

TOP_NN_VB_JJ=['wine', 'taste', 'warm', 'spice', 'red', 'dish', 'little', 'drink', 'pair', 'mead', 'perfect', 'holiday', 'try', ...]

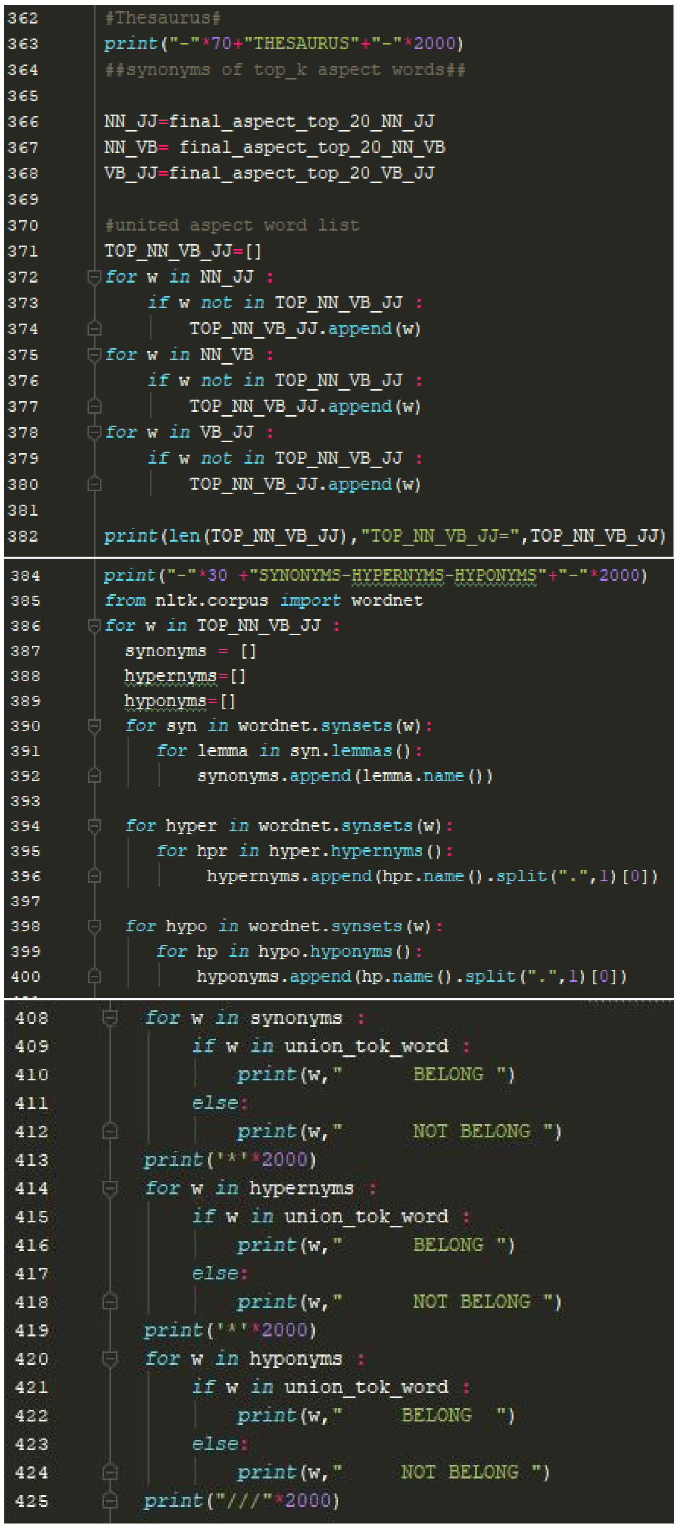
was extended into lists:

synonyms=['wine', 'vino', 'wine', 'wine-colored', 'wine-coloured', 'wine', 'wine']

hypernyms=['alcohol', 'dark_red', 'drink', 'regale']
hyponyms=['altar_wine', 'blush_wine', 'bordeaux', 'burgundy', 'dubonnet', 'california_wine', 'cotes_de_provence', 'dessert_wine',...]

Furthermore, something that we noticed was that an aspect word of the united aspect word list, has been also appeared many times in the list of *synonyms.*

Finally, in addition to extending aspects to lists containing synonyms, hypernyms, and hyponyms of them, we checked if these belong to the original unprocessed data (lines 408-424 in Figure 3.3). We observed that a lot of synonyms, hypernyms, and hyponyms of the aspect words have been included in the initial data and thus were also semantic words of the information retrieving. However, by providing detailed examples of applying the method, we will refer more details and observations for them in the next chapter.



```
#Thesaurus#
print("-"*70+"THESAURUS"+"-"*2000)
##synonyms of top_k aspect words##

NN_JJ=final_aspect_top_20_NN_JJ
NN_VB= final_aspect_top_20_NN_VB
VB_JJ=final_aspect_top_20_VB_JJ

#united aspect word list
TOP_NN_VB_JJ=[]
for w in NN_JJ :
    if w not in TOP_NN_VB_JJ :
        TOP_NN_VB_JJ.append(w)
for w in NN_VB :
    if w not in TOP_NN_VB_JJ :
        TOP_NN_VB_JJ.append(w)
for w in VB_JJ :
    if w not in TOP_NN_VB_JJ :
        TOP_NN_VB_JJ.append(w)

print(len(TOP_NN_VB_JJ),"TOP_NN_VB_JJ=",TOP_NN_VB_JJ)
```

```
print("-"*30 +"SYNONYMS-HYPERNYMS-HYPONYMS"+"-"*2000)
from nltk.corpus import wordnet
for w in TOP_NN_VB_JJ :
  synonyms = []
  hypernyms=[]
  hyponyms=[]
  for syn in wordnet.synsets(w):
      for lemma in syn.lemmas():
          synonyms.append(lemma.name())

  for hyper in wordnet.synsets(w):
      for hpr in hyper.hypernyms():
          hypernyms.append(hpr.name().split(".",1)[0])

  for hypo in wordnet.synsets(w):
      for hp in hypo.hyponyms():
          hyponyms.append(hp.name().split(".",1)[0])
```

```
  for w in synonyms :
      if w in union_tok_word :
          print(w,"      BELONG ")
      else:
          print(w,"      NOT BELONG ")
  print('*'*2000)
  for w in hypernyms :
      if w in union_tok_word :
          print(w,"      BELONG ")
      else:
          print(w,"      NOT BELONG ")
  print('*'*2000)
  for w in hyponyms :
      if w in union_tok_word :
          print(w,"     BELONG  ")
      else:
          print(w,"     NOT BELONG ")
  print("///"*2000)
```

Figure 3.3: Top aspects' expansion based on Thesauri.

Query's expansion based on Thesauri

In the previous paragraph, we referred to the last method of the system that we applied to our data. Before we analyze the data and query's connection, we should also describe the last method that has been applied to the user's query.

The query provided by the user is often unstructured and incomplete. An incomplete query hinders a search engine from satisfying the user's information need. In practice we need some representation which can correctly and more importantly completely express the user's information need. Thus, we decided to implement expansion of the user's query based on thesauri.

In particular, we tested if any of the query terms belonged to the top aspects' of the united aspect word list. If any term(s) belonged, then the question of the user would be extended to all the synonyms, hypernyms, hyponyms of the term(s) based on the *WordNet* of the *NLTK*, as before. Otherwise, if no query term was included in this list, then the query was remained as it was.

In this way, when at least one of the user's query terms belonged to the top aspect words, it was expanded into a better and more detailed form. This means that we transformed queries into a better layout from which we can obtain more accurate information while we could extract the most relevant documents of each domain at the same time. Hence, system's accuration has been increased and its users will be more satisfied.

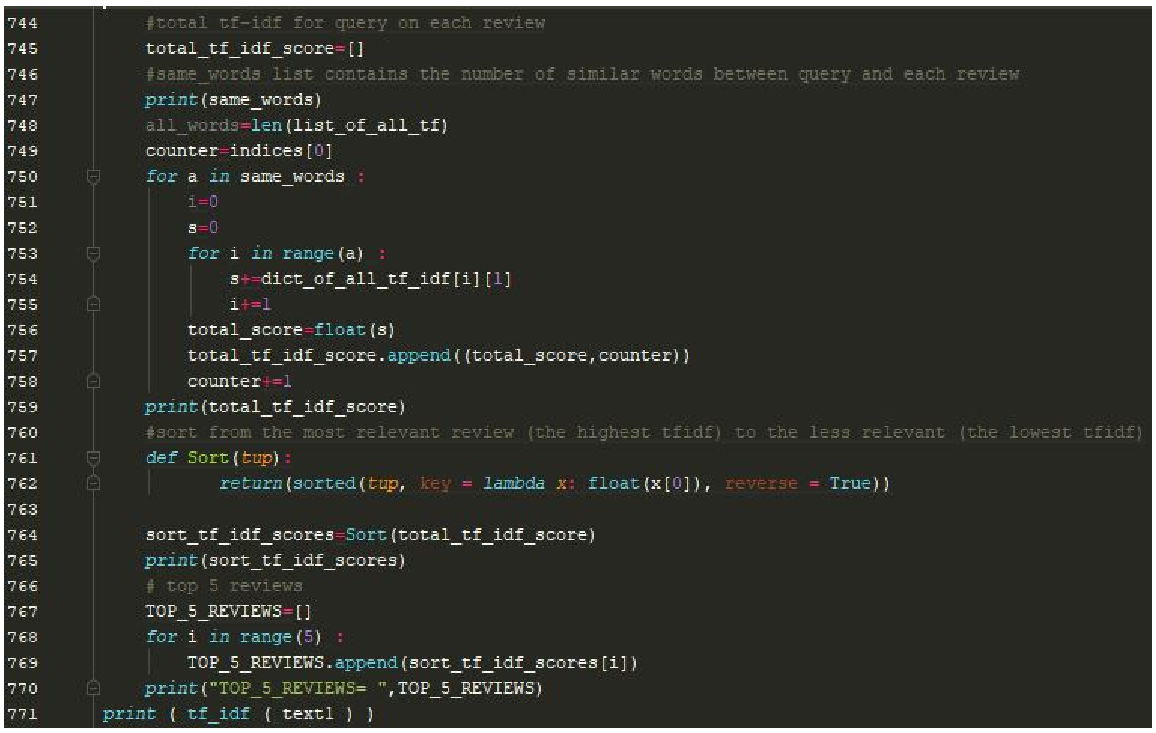
Connection between user's query and the most relevant documents

After describing all the procedures that have been applied to our data and the user's query separately it's time to describe how we combined these two parts in order to extract the most relevant documents to the query.

The statistical measure that we used for the retrievement of the most relevant documents to the query was that of Term Frequency-Inverse Document Frequency (TF-IDF) which has been analyzed in the Section 2.4. Specifically, the TF-IDF was contained to the score measure for a document given a query. This score is described below;

$$Score(q,d) = \sum_{t \in q \cap d} tf.idf_{t,d} \quad \textbf{(3.1)}$$

where $\boldsymbol{q}$ is the query (expanded or not) , $\boldsymbol{t}$ is the query term that is also



```
744     #total tf-idf for query on each review
745     total_tf_idf_score=[]
746     #same_words list contains the number of similar words between query and each review
747     print(same_words)
748     all_words=len(list_of_all_tf)
749     counter=indices[0]
750     for a in same_words :
751         i=0
752         s=0
753         for i in range(a) :
754             s+=dict_of_all_tf_idf[i][1]
755             i+=1
756         total_score=float(s)
757         total_tf_idf_score.append((total_score,counter))
758         counter+=1
759     print(total_tf_idf_score)
760     #sort from the most relevant review (the highest tfidf) to the less relevant (the lowest tfidf)
761     def Sort(tup):
762             return(sorted(tup, key = lambda x: float(x[0]), reverse = True))
763 
764     sort_tf_idf_scores=Sort(total_tf_idf_score)
765     print(sort_tf_idf_scores)
766     # top 5 reviews
767     TOP_5_REVIEWS=[]
768     for i in range(5) :
769         TOP_5_REVIEWS.append(sort_tf_idf_scores[i])
770     print("TOP_5_REVIEWS= ",TOP_5_REVIEWS)
771 print ( tf_idf ( text1 ) )
```

Figure 3.4: Score of a document given a query by using TF-IDF and the 5 most relevant documents to the query.

contained to the document, and ***d*** is a document.

Thus, at the first step of the method, we calculated TF-IDF measure for all the terms of the query that also have been contained at the documents. Then, we found the sum of the TF-IDF scores of the terms for each pair query-document (*Score(q,d)*) and we divided each *Score(q,d)* with the total number of the query terms that have been contained at all the documents of the dataset. Finally, we saved these Scores with the proportional index of the document to a list and we sorted it from the highest to the lowest Score. The first five items of the list are these with the highest Score and thus they represent the five most relevant documents to the user's query. Figure 3.4 (lines 744-771) above, contains a part of the algorithm of this method.

Chapter 4

Evaluation

Chapter 4 is essentially the testing and evaluation chapter of the system that we created by applying the methods mentioned in the previous chapter. Firstly, in the first two sections of it, the type and source of the data that has been used, as well as the experiments carried out are described. The final section contains a detailed representation of the results in tables and figures. In addition, the chapter contains, of course, a lot of comments about the results and generally about the system's accuracy, effectiveness, and flexibility.

4.1 Data description

The dataset that has been used for the conducting of the experiments has consisted of metadata, ratings and, product reviews from Amazon [**Snap.stanford.edu**, **2019**]. Metadata included descriptions, price, seles-rank, brand info, and co-purchasing links while ratings were not included metadata or reviews, but only (user, item, rating, timestamp) tuples. However, we used only the third category of data which included product reviews. Duplicate items of them have been removed and they have been sorted by product. Format was one-review-per-line in (loose) json. For further help reading there is an example below;

Sample Review of the domain : Reviews for Wines

"reviewerID"*: "AJFTIMKJ5BUR6",* ***"asin"****: "B001RTSMTO",* ***"reviewerName"****: "Kiera K̈iera"̈,* ***"helpful"****: [3, 3],* ***"reviewText"****: "I live near Quady, and have purchased this wine before. It is unique, it is fla-*

*vorful, it is delicious. It has notes of Rose Geranium - that tastes better than it sounds! Seriously, I love this wine. It is sweet, heady, and perfect for an after dinner drink with your significant other. I heartily recommend it.", **"overall"**: 5.0, **"summary"**: "Delicious!", **"unixReviewTime"**: 1354060800, **"reviewTime"**: "11 28, 2012"*

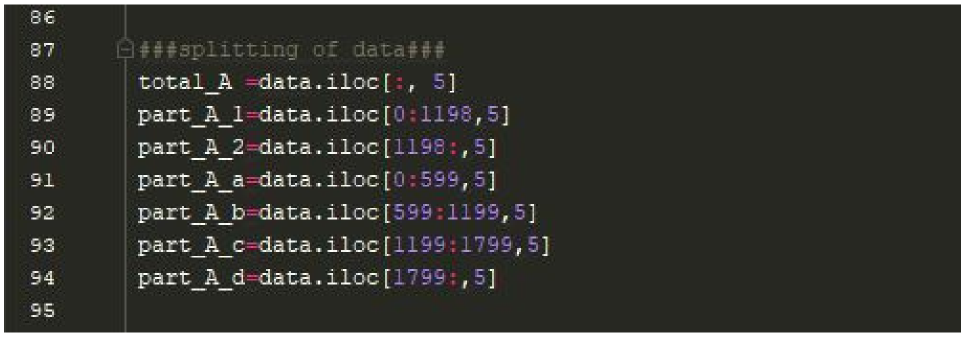
where:

- **"reviewerID"** -ID of the reviewer, e.g. AJFTIMKJ5BUR6.
- **"asin"** -ID of the product, e.g. "B001RTSMTO".
- **"reviewerName"** -Name of the reviewer.
- **"helpful"** -Helpfulness rating of the review, e.g. 3/3.
- **"reviewTextD"** - Text of the review.
- **"overall"** -Rating of the product.
- **"summary"** -Summary of the review.
- **"unixReviewTime"** -Time of the review (unix time).
- **"reviewTime"** -Time of the review (raw).

4.2 Experiments' description

In this section, have been recorded all the experiments that have been done in order to test the effectiveness of our Information Rrtrieval (IR) system. We used two domains of product reviews from Amazon : **A)** Reviews for wines and **B)** Reviews for baby products and by applying all the methods that have been referred to the previous chapter (e.g data and query's pre-processing, extraction of aspect words and expansion of them based on thesauri, expansion of the query, and extraction of the most relevant documents to the query) a total of 54 experiments were conducted. In addition to changing the two domains, we also changed the user's query and the reviews or the number of them.

More specifically, we started by conducting tests to the domain **A)**, which is consisted of 2396 reviews for wines. For the first fifteen tests, we used all the reviews (2396). Especially, we selected the text of the 2396 reviews of the dataset (line 88 of Figure 4.1). Thus, the only different part between



```
###splitting of data###
total_A =data.iloc[:, 5]
part_A_1=data.iloc[0:1198,5]
part_A_2=data.iloc[1198:,5]
part_A_a=data.iloc[0:599,5]
part_A_b=data.iloc[599:1199,5]
part_A_c=data.iloc[1199:1799,5]
part_A_d=data.iloc[1799:,5]
```

Figure 4.1: Splitting of the dataset A).

the first fifteen tests of domain A) was that of user's question. We refer the 15 different users' queries of the particular experiments below;

EXPERIMENTS

Except for the first question that the user generally searched for a bottle of good wine, the next six questions were more specific and the user searched for wines with many different aspects such as red or white, sweet or dry, cheap or expensive.

A_1) Query : "A bottle of good wine!"

A_2) Query : "A bottle of good and red wine!"

A_3) Query : "A bottle of good and sweet wine!"

A_4) Query : "A bottle of good, red, and sweet wine!"

A_5) Query : "A bottle of cheap, red, and sweet wine!"

A_6) Query : "A bottle of good, cheap, red, and sweet wine!"

A_7) Query : "A bottle of good, cheap, white, and dry wine!"

As a bottle of wine is combined very well with various menus (such as fish, a platter of different cheeses, etc.), the users' questions in the next five experiments were referred to the choice of wine that users have to make based on their choice of the menu.

A_8) Query : "What i have to choose for a pasta menu?"

A_9) Query : "What i have to choose for menu with many different cheeses?"

A_{10}) Query : "What i have to choose for a sushi menu?"

A_{11}) Query : "What i have to choose for menu with meat?"

A_{12}) Query : "What i have to choose for menu with fish?"

Finally, the last three experiments before we "break" the domain of the data of wines to two or more parts were included queries about specificl categories of wines (e.g champagne, cabernet, etc.). We refer the questions of the trials below;

A_{13}) Query : "A champagne for the celebration!"

A_{14}) Query : "A bottle of Cabernet!"

A_{15}) Query : "What about a Sauvignon Blanc?"

After the conduction of the 15 trials, we splitted the dataset first into two parts (lines 89-90 in Figure 4.1) and we repeated some of the previous questions. Thus, ten experiments were carried out after the two splittings of the dataset and together with the previous ones are shown in the Table 4.1.

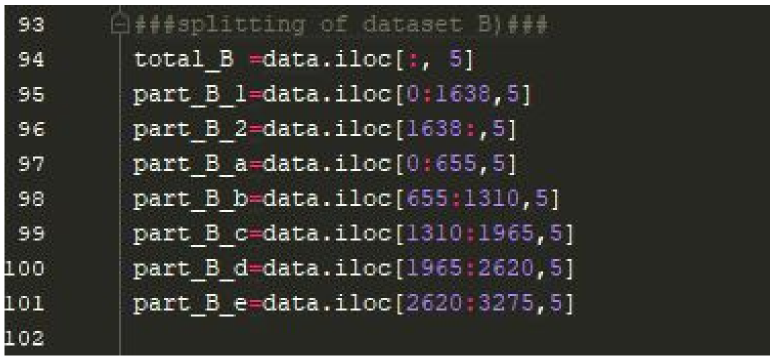
For example the fifth row,

$$(A_4, \text{"A bottle of good, red, and sweet wine!"}, \textit{total\_A})$$

of the Table 4.1 is referred to the fourth experiment of the domain A) in which user wants to find the most relevant documents to his/her query (" A bottle of good, red, and sweet wine!") between the 2396 documents of the whole domain, while the eighteenth row,

$$(A_{17}, \text{ "A bottle of good wine!"}, \textit{part\_A\_2} \ )$$

is referred to the seventeenth trial, in which user's goal is the extraction of the most relevant documents to his/her query ("A bottle of good wine!") between the 1999 to 2396 documents of the domain.



```
93   ###splitting of dataset B)###
94   total_B =data.iloc[:, 5]
95   part_B_1=data.iloc[0:1638,5]
96   part_B_2=data.iloc[1638:,5]
97   part_B_a=data.iloc[0:655,5]
98   part_B_b=data.iloc[655:1310,5]
99   part_B_c=data.iloc[1310:1965,5]
100  part_B_d=data.iloc[1965:2620,5]
101  part_B_e=data.iloc[2620:3275,5]
102
```

Figure 4.2: Splitting of the dataset B).

Correspondingly, we summarized the experiments that have been conducted to the domain **B)** of the reviews of products for babies in the Table A.1 (Appendix A). Domain B) is consisted of 3275 reviews for baby products and, as before, by using *panda* library in Python we selected the *reviewText* of the 3275 reviews of the dataset (line 94 in Figure 4.2) and we used them for the first fifteen tests. As a result, the only part that changed between these tests was that of user's question.

Finally, we splitted the dataset for the next 7 experiments of domain B). At first, the division of it was into two parts (lines 95-96 of Figure 4.2), while the second splitting was at five parts (lines 97-101 of Figure 4.2) and we repeated two of the questions that have been used on the first fifteen tests of domain B).

Experiment's Name	**User's Query**	**Reviews**
A_1	"A bottle of good wine!"	*total_A*
A_2	"A bottle of good and red wine!"	*total_A*
A_3	"A bottle of good and sweet wine!"	*total_A*
A_4	"A bottle of good, red, and sweet wine!"	*total_A*
A_5	"A bottle of cheap, red, and sweet wine!"	*total_A*
A_6	"A bottle of good, cheap, red, and sweet wine!"	*total_A*
A_7	"A bottle of good, cheap, white, and dry wine!"	*total_A*
A_8	"What i have to choose for a pasta menu?"	*total_A*
A_9	"What i have to choose for menu with many different cheeses?"	*total_A*
A_{10}	"What i have to choose for a sushi menu?	*total_A*
A_{11}	"What i have to choose for menu with meat ?"	*total_A*
A_{12}	"What i have to choose for menu with fish?"	*total_A*
A_{13}	"A champagne for the celebration!"	*total_A*

A_{14}	"A bottle of Cabernet!"	*total_A*
A_{15}	"What about a Sauvignon Blanc?"	*total_A*
A_{16}	"A bottle of good wine!"	*part_A_1*
A_{17}	"A bottle of good wine!"	*part_A_2*
A_{18}	"A bottle of good wine!"	*part_A_a*
A_{19}	"A bottle of good wine!"	*part_A_b*
A_{20}	"A bottle of good wine!"	*part_A_c*
A_{21}	"A bottle of good wine!"	*part_A_d*
A_{22}	"A bottle of good, cheap, red, and sweet wine!"	*part_A_1*
A_{23}	"A bottle of good, cheap, red, and sweet wine!"	*part_A_2*
A_{24}	"A bottle of good, cheap, white, and dry wine!"	*part_A_1*
A_{25}	"A bottle of good, cheap, white, and dry wine!"	*part_A_2*

Table 4.1: Experiments of the domain A).

4.3 Results' representation and analysis

In this section, are reported the findings of my study based upon the methodologies that been applied in the different experiments of the previous section in order to gather information. Specifically, this section of the results contains the findings of the research that have been arranged based on a logical sequence of the methods that have been applied (1) data preprocessing, 2) aspects' extraction and expansion of them on thesauri 3) query system for the retrieving of the most relevant reviews). In addition, after each table of results there are observations and analysis of them.

DATA PRE-PROCESSING

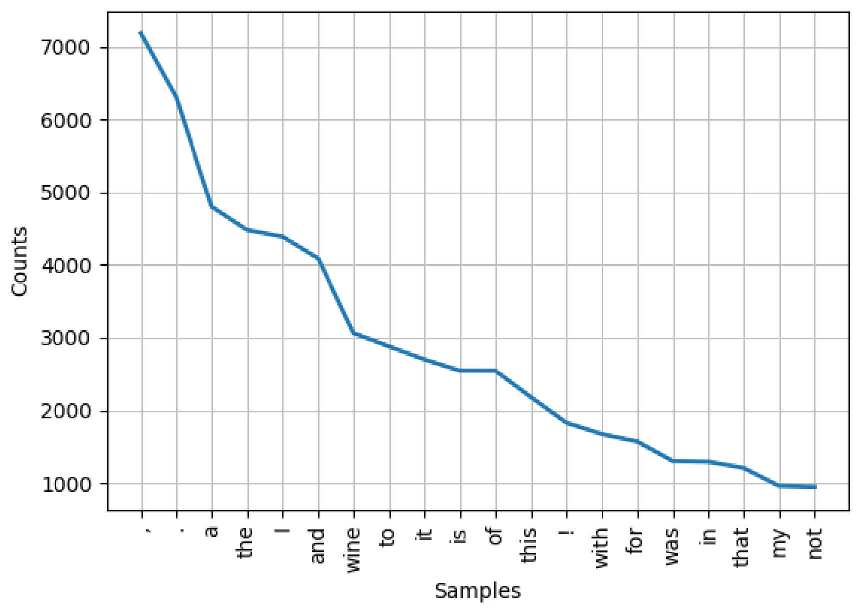
Experiment	**Reviews**	**Top_20 words before pre-processing (b.p)**	**Top_20 words after pre-processing (a.p)**	**Number of words of the union of reviews (b.p) and (a.p)**
$A_1 - A_{15}$	*total_A*	[',', '.', 'a', 'the', 'I', 'and', 'wine', 'to', 'it', 'is', 'of', 'this', '!', 'with', 'for', 'was', 'in', 'that', 'my', 'not']	['wine', 'taste', 'bottle', 'win', 'like', 'great', 'good', 'love', 'try', 'buy', 'drink', 'sweet', 'enjoy', 'red', 'flavor', 'price', 'well', 'really', 'order', 'gift']	(b.p)=155.316 (a.p)=66.899
A_{16}, A_{22}, A_{24}	*part_A_1*	[',' , '.', 'a', 'I', 'the', 'and', 'wine', 'it', 'to', 'is', 'of', 'this', 'with', '!', 'for', 'that', 'in', 'was', 'but', 'not']	['wine', 'taste', 'like', 'bottle', 'win', 'great', 'good', 'try', 'love', 'buy', 'drink', 'sweet', 'red', 'flavor', 'enjoy', 'price', 'well', 'smooth', 'go', 'make']	(b.p)=85.647 (a.p)=37.046
A_{17}, A_{23}, A_{25}	*part_A_2*	[',', '.', 'the', 'a', 'I', 'and', 'to', 'wine', 'of', 'is', 'it', 'this', '!', 'with', 'for', 'was', 'in', 'that', 'my', 'not']	['wine', 'great', 'win', 'taste', 'bottle', 'love', 'like', 'good', 'buy', 'try', 'enjoy', 'drink', 'gift', 'red', 'pinot', 'really', 'order', 'sweet', 'flavor', 'price']	(b.p)=68.754 (a.p)=29.853
A_{18}	*part_A_a*	[',', '.', 'a', 'I', 'the', 'and', 'wine', 'it', 'to', 'is', 'of', 'this', 'with', 'for', '!', 'that', 'in', 'was', 'you', 'not']	['wine', 'taste', 'like', 'bottle', 'win', 'good', 'try', 'drink', 'sweet', 'great', 'buy', 'love', 'red', 'price', 'flavor', 'enjoy', 'serve', 'go', 'well', 'smooth']	(b.p)=47.712 (a.p)=20.455

A_{19}	*part_A_b*	[',', '.', 'a', 'the', 'I', 'and','wine', 'to', 'it', 'is', 'of', 'this', '!', 'with', 'for', 'in', 'was', 'that', 'but', 'not']	['wine', 'taste', 'bottle', 'like', 'great', 'love', 'win', 'good', 'buy', 'try', 'sweet', 'drink', 'flavor', 'enjoy', 'red', 'well', 'really', 'price', 'recommend', 'smooth']	(b.p)=37.984 (a.p)=16.614
A_{20}	*part_A_c*	[',', '.', 'a', 'the', 'I', 'and', 'wine', 'to', 'of', 'is', 'it', '!', 'this', 'with', 'for', 'was', 'in', 'that', 'my', 'but']	['wine', 'great', 'win', 'taste', 'like', 'bottle', 'love', 'good', 'buy', 'try', 'pinot', 'enjoy', 'drink', 'flavor', 'red', 'gift', 'nice', 'give', 'really', 'm']	(b.p)=35.762 (a.p)=15.634
A_{21}	*part_A_d*	[',', '.', 'the', 'a', 'I', 'and', 'to', 'wine', 'of', 'it', 'is', 'this', '!', 'for', 'with', 'was', 'in', 'that', 'have', 'my']	['wine', 'win', 'great', 'taste', 'bottle', 'good', 'love', 'like', 'buy', 'try', 'enjoy', 'order', 'gift', 'foxen', 'drink', '34', 'price', 'really', 'red', 'sweet']	(b.p)=32.943 (a.p)=14.196

Table 4.2: Results of the pre-processing of data of the domain A).

First of all, by observing the last column of Table 4.2 above, we noticed that after the preprocessing of the reviews, the number of their words has decreased significantly (more than half). Therefore, while punctuation remarks and stopwords were primarily the most frequent terms and have been counted more times in the text of product reviews (as we can see in Figure 4.3 below), after the procedure of the preprocessing, terms that express an opinion or aspects of the products now constitute the words with the highest frequency.

An additional observation for Table 4.2 is that the 4th column does not change significantly during the 25 experiments. This means that we can use even less number of reviews to find the most common words that appear in the dataset and thus we will save time and resources.



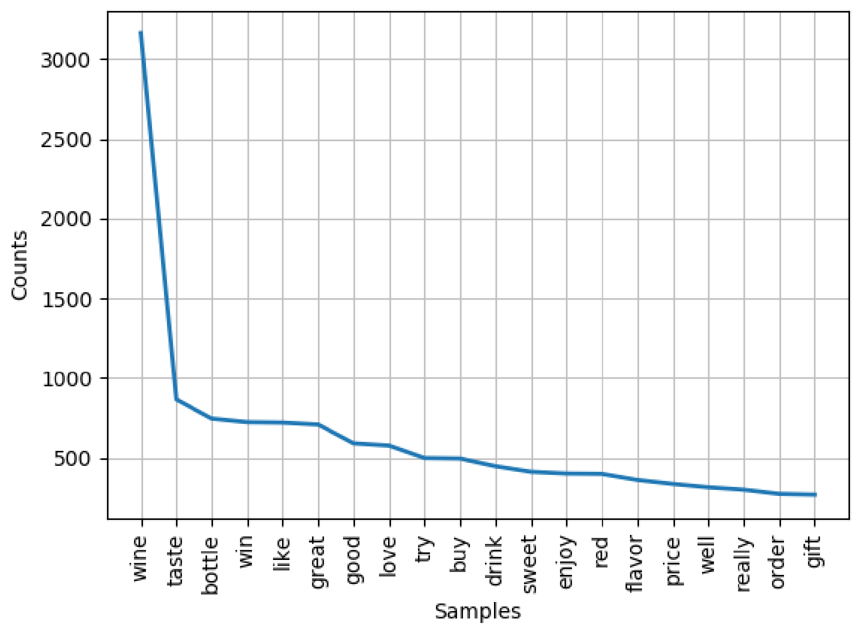


Figure 4.3: Twenty most frequent words of the first fifteen experiments of domain A) before(up) an after(down) pre-processing.

ASPECTS' EXTRACTION AND EXPANSION OF THEM ON THESAURI

Experiment	**Reviews**	**Top-aspects (TOP_NN_VB_JJ)**	**Thesauri of top aspects**
$A_1 - A_{15}$	*total_A*	['wine', 'great', 'good', 'bottle', 'taste', 'sweet', 'red', 'price', 'nice', 'pinot', 'favorite', 'flavor', 'gift', 'fruit', 'dinner', 'delicious', 'perfect', 'try', 'love', 'drink', 'glass', 'cabernet', 'order', 'enjoy', 'food', 'white', 'dry', 'smooth', 'little',...]	Example of aspect : wine **synonyms_of_wine**=['wine', 'vino', 'wine-colored', 'wine-coloured"] **hypernyms_of_wine**=['alcohol', 'dark_red', 'drink', 'regale',...] **hyponyms_of_wine**=['bordeaux', 'burgundy', 'tokay', 'varietal', 'vermouth', 'vintage',...]
A_{16}, A_{22}, A_{24}	*part_A_1*	['wine', 'great', 'good', 'bottle', 'taste', 'sweet', 'red', 'price', 'flavor', 'nice', 'favorite', 'smooth', 'dinner', 'white', 'fruit', 'delicious', 'love', 'glass', 'cabernet', 'try', 'drink', 'fruity', 'merlot', 'pinot', 'gift', 'food', 'little', 'perfect', 'dry',...]	Example of aspect : fruit **synonyms_of_fruit**=['fruit', 'yield'] **hypernyms_of_fruit**=['product', 'consequence', 'bear'] **hyponyms_of_fruit**= ['achene', 'acorn', 'berry', 'buckthorn_berry', 'chokecherry', 'cubeb', 'drupe',...]
A_{17}, A_{23}, A_{25}	*part_A_2*	['wine', 'great', 'good', 'bottle', 'red', 'pinot', 'taste', 'sweet', 'gift', 'nice', 'price', 'favorite', 'winery', 'fruit', 'flavor', 'dry', 'delicious', 'order', 'dinner', 'love', 'try', 'buy', 'drink', 'year', 'family', 'enjoy', 'white', 'perfect', 'excellent', 'new',...]	Example of aspect : delicious **synonyms_of_delicious**=['delightful', 'yummy', 'delicious', 'delectable', 'luscious',...] **hypernyms_of_delicious**=['eating_apple'] **hyponyms_of_delicious**= ['golden_delicious', 'red_delicious']
A_{18}	*part_A_a*	['wine', 'good', 'bottle', 'great', 'taste', 'sweet', 'red', 'price', 'flavor', 'white', 'smooth', 'glass', 'fruit', 'dry', 'cabernet', 'delicious', 'dinner', 'drink', 'try', 'buy', 'love', 'food', 'merlot', 'fruity', 'pinot', 'gift', 'perfect', 'local',...]	Example of aspect : red **synonyms_of_red**=['red', 'redness', 'reddish', 'ruddy', 'cerise', 'cherry', 'crimson', 'ruby',...] **hypernyms_of_red**=['sum', 'radical', 'chromatic_color'] **hyponyms_of_red**= ['cerise', 'chrome_red', 'crimson', 'dark_red', 'purplish_red', 'sanguine']

A_{19}	*part_A_b*	['wine', 'great', 'bottle', 'good', 'taste', 'sweet','finish' 'red', 'price', 'winery', 'flavor', 'love', 'nice', 'dinner', 'pinot', 'delicious', 'smooth', 'fruit', 'try', 'drink', 'cabernet', 'gift', 'glass', 'chocolate', 'fruity', 'enjoy', 'perfect', 'white', 'rich', 'big',...]	Example of aspect : taste **synonyms_of_taste**=['taste', 'try', 'preference', 'penchant', 'savor'] **hypernyms_of_taste**=['sensation', 'experience', 'exteroception', 'modality', 'sensing',...'] **hyponyms_of_taste**=['bitter', 'finish', 'mellowness', 'relish', 'salt', 'sour', 'sweet']
A_{20}	*part_A_c*	['wine', 'great', 'good', 'pinot', 'taste', 'nice', 'sweet', 'favorite', 'gift', 'flavor', 'fruit', 'price', 'winery', 'dinner', 'excellent', 'try', 'love', 'drink', 'family', 'buy', 'glass', 'order', 'perfect', 'white', 'different', 'recommend', 'finish', 'dry',...]	Example of aspect : recommend **synonyms_of_recommend**=['urge', 'advocate', 'commend','recommend'] **hypernyms_of_recommend**=['praise', 'propose', 'change'] **hyponyms_of_recommend**=[ ]
A_{21}	*part_A_d*	['wine', 'great', 'good', 'bottle', 'sweet', 'foxen', 'red', 'taste', 'gift', 'price', 'nice', 'favorite', 'winery', 'order', 'perfect', 'dry', 'love', 'try', 'buy', 'drink', 'enjoy', 'dinner', 'year', 'fruit', 'flavor', 'club', 'white', 'new', 'little', 'wonderful',...]	Example of aspect : drink **synonyms_of_drink**=['boozing', 'beverage','potable', 'drink', 'swallow', 'toast', 'pledge', 'salute'] **hypernyms_of_drink**=['food', 'liquid'] **hyponyms_of_drink**=['draft', 'nightcap', 'sangaree', alcohol', 'cider', 'cocoa', 'coffee', 'gulp', 'oenomel', 'wine',...]

Table 4.3: Results of the aspects' extraction and expansion of them on thesauri of the domain A).

In addition, something similar to the second observation of Table 4.2 was observed in the 3rd column of Table 4.3 above. Regardless of the experiment we conducted and therefore regardless of the number of reviews, the set of aspect words that we extracted in each test is almost always the same. Only the order of the aspects of the set was changed (due to the fact that they are placed in descending frequency) and a new aspect was rarely added to it. Hence, it's a good idea to use parts of the dataset for the extraction of aspect words as this will improve cost-effectiveness and time-efficiency of our system.

However, the most significant column of Table 4.3 is the last one. This contains examples of the expansion of the aspect words to their synonyms, hypernyms, and hyponyms and many of these thesauri-words belonged to the initial reviews (e.g the word “dessert” and “fresh” which are synonyms of the term “sweet” and “taste” which is a hypernym of it). This fact leads us to the user’s query expansion when at least one of the query belongs to the set of aspects, as many of these questions can be made in many different ways!

QUERY SYSTEM FOR THE RETRIEVING OF THE MOST RELEVANT REVIEWS

Exp.	**Preprocessed Query**	**Expansion of Query**	**Score_tf-idf and 5 most Relevant Reviews (R)**	**Score and 5 most Rel. Rev. (R) without expansion**
A_1	['bottle', 'good', 'wine']	*YES*	(0.45951, R:72) (0.45951, R:1009) (0.30111, R:263) (0.30111, R:274) (0.30111, R:364)	(0.01687, R:5) (0.01687, R:17) (0.01687, R:18) (0.01687, R:24) (0.01687, R:34)
A_2	['bottle', 'good', 'red', 'wine']	*YES*	(0.49126, R:72) (0.47603, R:1009) (0.27338, R:263) (0.27338, R:364) (0.27338, R:841)	(0.05856, R:100) (0.05856, R:152) (0.05856, R:263) (0.05856, R:265) (0.05856, R:273)
A_3	['bottle', 'good', 'sweet', 'wine']	*YES*	(0.72140, R:81) (0.66291, R:1058) (0.57089, R:72) (0.57089, R:1057) (0.57089, R:1102)	(0.06573, R:34) (0.06573, R:65) (0.06573, R:135) (0.06573, R:210) (0.06573, R:233)
A_4	['bottle', 'good', 'red', 'sweet', 'wine']	*YES*	(0.67943, R:72) (0.67943, R:81) (0.58742, R:1058) (0.56146, R:1057) (0.56146, R:1102)	(0.08226, R:310) (0.06552, R:9) (0.06552, R:34) (0.06552, R:64) (0.06552, R:65)
A_5	['bottle', 'cheap', 'red', 'sweet', 'wine']	*YES*	(0.56971, R:81) (0.51122, R:180) (0.41921, R:72) (0.41921, R:274) (0.41921, R:1057)	(0.08226, R:492) (0.06552, R:9) (0.06552, R:180) (0.06552, R:200) (0.06552, R:225)
A_6	['bottle', 'good', 'cheap', 'red', 'sweet', 'wine']	*YES*	(0.67943, R:72) (0.67943, R:81) (0.58742, R:1058) (0.56146, R:180) (0.56146, R:274)	(0.082226, R:310) (0.082226, R:492) (0.08226, R:829) (0.0.08226, R:2190) (0.08226, R:9)
A_7	['bottle', 'good', 'cheap', 'white', 'dry', 'wine']	*YES*	(0.51807, R:72) (0.51807, R:1009) (0.35967, R:274) (0.31542, R:263) (0.31542, R:1057)	(0.10060, R:2190) (0.07544, R:17) (0.07544, R:72) (0.07544, R:79) (0.07544, R:135)

A_8	['choose', 'pasta', 'menu']	*NO*	(0.02582, R:1) (0.02582, R:40) (0.02582, R:72) (0.02582, R:86) (0.02582, R:117)
A_9	['choose', 'menu', 'many', 'different', 'cheese']	*NO*	(0.05019, R:10) (0.05019, R:72) (0.05019, R:79) (0.05019, R:207) (0.05019, R:211)
A_{10}	['choose', 'sushi', 'menu']	*NO*	(0.07965, R:40) (0.04527, R:117) (0.04527, R:125) (0.04527, R:243) (0.04527, R:310)
A_{11}	['choose', 'menu', 'meat']	*NO*	(0.02484, R:1) (0.02484, R:40) (0.02484, R:80) (0.02484, R:117) (0.02484, R:125)
A_{12}	['choose', 'menu', 'fish']	*NO*	(0.02736, R:1) (0.02736, R:40) (0.02736, R:72) (0.02736, R:105) (0.02736, R:117)
A_{13}	['champagne', 'celebration']	*NO*	(0.03772, R:23) (0.001257, R:58) (0.03772, R:70) (0.03772, R:92) (0.03772, R:139)
A_{14}	['bottle', 'cabernet']	*YES*	(0.04699, R:351) (0.04044, R:65) (0.04044, R:66) (0.04044, R:73) (0.04044, R:79)
A_{15}	['sauvignon', 'blanc']	*NO*	(0.16332, R:25) (0.16332, R:58) (0.16332, R:75) (0.16332, R:139) (0.16332, R:221)
A_{16}	['bottle', 'good', 'wine']	*YES*	(0.42717, R:72) (0.42717, R:1009) (0.28758, R:263) (0.28758, R:274) (0.28758, R:364)
A_{17}	['bottle', 'good', 'wine']	*YES*	(0.20844, R:2190) (0.20154, R:2045) (0.18179, R:2169) (0.18179, R:2175) (0.17778, R:1206)

A_{18}	['bottle', 'good', 'wine']	*YES*	(0.39454, R:72) (0.39454, R:364) (0.27377, R:81) (0.27377, R:93) (0.27377, R:263)
A_{19}	['bottle', 'good', 'wine']	*YES*	(0.46577, R:1009) (0.43194, R:1102) (0.42535, R:1058) (0.42535, R:1065) (0.40235, R:753)
A_{20}	['bottle', 'good', 'wine']	*YES*	(0.17363, R:1206) (0.17363, R:1228) (0.17363, R:1271) (0.17363, R:1280) (0.17363, R:1465)
A_{21}	['bottle', 'good', 'wine']	*YES*	(0.31464, R:2190) (0.28843, R:2045) (0.23704, R:2169) (0.23704, R:2175) (0.22165, R:2376)
A_{22}	['bottle', 'good', 'cheap', 'red', 'sweet', 'wine']	*YES*	(0.63379, R:72) (0.63379, R:81) (0.54678, R:1058) (0.52069, R:180) (0.52069, R:274)
A_{23}	['bottle', 'good', 'cheap', 'red', 'sweet', 'wine']	*YES*	(0.48505, R:2190) (0.30347, R:1396) (0.30347, R:1541) (0.26938, R:1206) (0.26938, R:1520)
A_{24}	['bottle', 'good', 'cheap', 'white', 'dry', 'wine']	*YES*	(0.48505, R:72) (0.34546, R:1009) (0.34546, R:274) (0.30454, R:263) (0.30454, R:1057)
A_{25}	['bottle', 'good', 'cheap', 'white', 'dry', 'wine']	*YES*	(0.28225, R:2190) (0.20844, R:1280) (0.20844, R:2045) (0.20154, R:1575) (0.20154, R:2175)

Table 4.4: Query system and the five most relevant reviews to user's query of the experiments on domain A).

The most useful results of the query system are summarized to the last table of chapter 4.3 (Table 4.4). The 2nd and 3rd columns of this table are represent the pre-processed user's query and if it was expanded (YES) or not (NO), respectively, while the 4th column is this which deserves more explanation and disccusion. Also, for the first 7 experiments there is an extra column with the Score(q,R) and the five most relevant reviews to the query in case that we don't take the expanded form of the query.

The first number of each pair in the 4th column represents the Score(q,R), while the second one, is the number of the relevant review R.

For example, the most relevant review to the expanded query of experiment A_25 is the 2190th review(R) with Score(q,R)=0.000127.

Thus, we observed reviews that are relevant to many queries (e.g the 72nd review is relevant to the expanded queries of the experiments A_1-A_9 and A_12 and A_15) while there are some others that are relevant only to one(e.g the 1065th review is relevant only to the expanded query of A_2, the 10th to the query of A_9, 80th to the query of A_11, etc.). One more significant notice is that the most relevant documents are primarily found in the first half of the dataset. Finally, there are pairs of documents which are at the same time relevant to two or more queries (e.g the 139th and 58th reviews are relevant to the queries of the experiments A_13 and A_15 at the same time). Probably, this happens because the pairs of documents, simultaneously, have relevant information to the both questions.

Finally, by comparing the 4th and 5th columns of the first 7 experiments we can notice that the Score(q,R) of the five most relevant reviews is much higher when we select the expanded form of the query and these reviews are totally different of the five most relevant reviews to the not-expanded query. This fact leads us to think that it is almost necessary to extend the question in order to obtain more accurate results.

Chapter 5

Conclusion and future work

The purpose of this research was the design and development of an intelligent information retrieval system. More specifically, a system based on short users' queries to extract the most relevant documents that contain useful information and express opinion. For this goal, many different methods have been combined in order to succesfully design a time-efficient, cost-effective and intelligent IR system, such as the pre-processing of data and user's query which contributed to the reduction of the volume of the data and to the transformation of questions in the latter of keywords, respectively. Other significant methodologies as defined in Chapter 3 were that of aspects' extraction and the expansion of user's query based on the thesauri of these aspects. As it was proven on the experiments (Chapter 4), the extraction of aspect words was an essential procedure for the increase of the accuracy of our system, as most information and opinions are gathered on these words. Moreover, the expansion of the queries to synonyms, hypernyms, or hyponyms of the aspect words turned out necessary, as many times an individual is searching for a product or an idea with specific aspects and expresses it on an equivalent way.

As a result, all these methods have been used in different experiments to evaluate our system (Chapter 4). Indeed, in these experiments, the purpose of the thesis was completed by using the statistical measure tf-idf and therefore we extracted the most relevant documents to the different questions that users inserted to the system. The experiments resulted in various observations and conclusions as mentioned in section 4.3, with the most significant to be that some documents are related to more than one question. This is also the reason that motivated me to think about future work and enhancement of my system with further methods. Hence, a future system could also check beyond relevance between query and data, the relevance between the queries. Finally, one more good thought would be to take advantage of further elements of the documents/reviews (e.g “helpful” or “reviewTime”) in order to expand our system with a method that will generate results ranked by other criteria (e.g reliability or date).

Bibliography

[1] Chaudhuri, A.,2006. *Emotion and reason in consumer behavior.* USA: Routledge Taylor & Francis Group.

[2] Grefenstette, G.,1994. *Explorations in Automatic Thesaurus Discovery.*Boston-London-Dordrecht : Kluwer Academic Publishers.

[3] Karen, Spark J.,1986. *Synonymy and Semantic Classification.* Edinburgh, Scotland: Edinburgh University Press.

[4] Liu, B., 2015. *Sentiment Analysis : Mining Opinions, Sentiments and Emotions.* USA : Cambridge University Press.

[5] Manning, D. C., Raghavan, P. and Schutze, H., 2009. *Introduction to Information Retrieval.* Cambridge, England : Cambridge University Press.

[6] Sarawagi, S.,2008. *Information Extraction : Foundations and Trends in Databases.*USA: Publishers Inc.

[7] Hearst, M.,1992. *Direction-Based Text Interpretation as an Information Access Refinement.* From Text-Based Intelligent Systems, edited by Paul Jacobs, Lawrence Erlbaum Associates:257-274.

[8] Liu, B.,2010. *Sentiment Analysis and Subjectivity.*In Handbook of N. Indurkhya and F. J. Damerau Natural Language Processing. London: CRC Press:1-58.

[9] Morik, K. and Scholz, M.,2004. *The Mining Mart Approach to Knowledge Discovery in Databases.* In book of Ning Zhong and Jimming Liu :Intelligent Technologies for Information Analysis, Verlag Berlin Heidelberg : Springer:47-65.

[10] Zhang, L. and Liu, B.,2014.*Aspect and Entity Extraction for Opinion Mining.* in book of Wesley W. Chu : Data Mining and Knowledge Discovery for Big Data, Verlag Berlin Heidelberg : Springer:1-40.

[11] Abulaish, M., Jahiruddin, Doja, M. N. and Ahmad, T.,2009. Feature and Opinion Mining for Customer Review Summarization. *PReMI 2009.*,pp.219-224.

[12] Banerjee, S. and Pedersen, T.,2002. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. *Proceedings of the Inrternational Conference on Intelligent Text Processing and Computational Linguistics*,pp.136-145.

[13] Belinkov, Y., Lei, T., Barzilay, R. and Globerson, A.,2014. Exploring Compositional Architectures and Word Vector Representations for Prepositional Phrase Attachment. *Transactions of the Association for Computational Linguistics 2*,pp.561-572.

[14] Blei, D., Y.Ng, A. and Jordan, M.,2003. Latent Dirichlet Allocation. *The Journal of Machine Learning.*,pp.993-1022.

[15] Bloom, K., Garg, N. and Aryamon, S.,2007. Extracting Appraisal Expressions.*In proceedings of the 2007 Annual Conference of the North American Chapter of the ACL(NAACL-2007).*

[16] Budanitsky, A. and Hirst, G.,2001. Semantic distance in WordNet : An experimental, application-orriented evaluation of five measures. *In proceedings of NAACL 2001 WordNet and other Lexical Resources Workshop.*,pp.29-34.

[17] Ceron, A., Curini, L. and Iacus, S. M.,2012. Tweet your vote: How content analysis of social networks canimprove our knowledge of citizens' policy preferences. An application to Italy and France. *Presented at the XXVI Congress of Italian Society of Political Science (SISP).*,pp.340-358.

[18] Das, S. and Chen, M.,2004. Yahoo! for Amazon: Sentiment Extraction from Small Talk on the Web. *Paper presented at the Asia Pasific Finance Association Annual Conference*,pp.1-16.

[19] Dave, K., Lawrence, S. and Pennock, D.M.,2003. Mining the Peanut Gallery : Opinion Extraction and Semantic Classification of Product Review.*In proceedings of International Conference on World Wide Web(WWW-2003).*

[20] Farooq, U.,Mansoor, H., Nongaillard, A., Ouzrout, Y. and Qadir, M.,2016. Negation Handling in Sentiment Analysis at Sentence Level,*In Journal of Computers.*

[21] Feldman, R., Fresco, M., Goldenberg, J., Netzer, O. and Ungar, L.,2007. Extracting Product Comparisons from Discussion Boards *Seventh IEEE International Conference on Data Mining*,pp.469-474.

[22] Goeuriot, L., Na, J., Kyaing, W., Foo, S., Khoo, C., Theng, Y. and Chang, Y.,2011. Textual and Informational Characteristics of Health-Related Social Media Content: A Study of Drug Review Forums. *Asia Pacific Conference Library & Information Education & Practice.*,pp. 548-557.

[23] Grosz, B., Joshi, A. and Weinstein, S.,1995. Centering: A Framework for Modeling the Local Coherence of Discourse.*Computational Linguistics*,21(2),pp.205-225.

[24] Hatzivassiloglou, V. and McKeown , C.,1997. Predicting the Semantic Orientation of Adjectives . *In Proceedings of Annual Meeting of the Association for Computational Linguistics(ACL-1997)*,pp.174-181.

[25] Hou, Y.,2018. A Deterministic Algorithm for Bridging Anaphora Resolution. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.*,pp.1938-1948.

[26] Hu, M. and Liu, B.,2004. Mining and Summarizing Customer Reviews. *In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

[27] Hu, M. and Liu, B.,2004a. Mining Opinion Features in Customer Reviews. *In Proceedings of Nationall Conference on Artificial Intelligence.(AAAI-2004.*,pp.755-760.

[28] Jana, A. and Goyal, P.,2018. Network Features Based Co-hyponymy Detection. *Indian Institute of Technology Kharagpur.*

[29] Jing, Y. and Croft, W.,1994. An Association Thesaurus for Information RetrievaL. *In proceedings at RIAO '94 Intelligent Multimedia Information Retrieval Systems and Management.*,pp.146-160.

[30] Kalaria, A. and Prajapati, Z.,2016. Opinion Mining for Information Retrieval: Survey *International Journal of Computer Science and Network.*,5[6],pp.934-940.

[31] Kessler, J. S., Eckert, M., Clark, L., and Nicolov, N.,2010. The ICWSM 2010 JDPA Sentiment Corpus for the Automotive Domain. *Paper presented at the 4th International AAAI Conference on Weblogs and Social Media Data Workshop.*

[32] Khan, K., Baharudin, B., Khan, A., and Ullah, A.,2014. Mining opinion components from unstructures reviews :A review. *Journal of King Saud University-Computer and Information Sciences.*,26,pp.258-275.

[33] Kilgarrif, A.,2003. Thesauruses for Natural Language Processing.*In proceedings of the International Conference on Natural Language Processing and Knowledge Emgineering.*

[34] Kummerfeld, J., Hall, D. and Klein, D.,2012. Parser Showdown at the Wall Street Corral: An Empirical Investigation of Error Types in Parser Output.*Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning.*,pp.1048-1059.

[35] Lafferty, J., McCallum, A. and Pereira, F.,2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *In proceedings of the 18th International Conference on Machine Learning(ICML-2001)*,pp.282-289.

[36] Lin, D.,1998. Automatic Retrieval and Clustering of Similar Words.*COLING-ACL*,pp.768-774.

[37] Lin, C. and Chao, P.,2010. Tourism-Related Opinion Detection and Tourist-Attraction Target Identification.*Int J. Computational Linguistics and Chinese Language Processing.*,pp.37-60.

[38] Liu, B.,2007. Opinion Mining, pp.1-7.

[39] Lu, Y., Duan, H., Wang, H. and Zhai C.,2010. Exploiting Structured Ontology to Organize Scattered Online Opinions. *In Proceedings of International Conference of Computational Linguistics (COLING-2010).*,pp.734-742.

[40] Ma, T. and Wang, X., 2010. Opinion Target Extraction in Chinese News Comments.*In Proceedings of International Conference of Computational Linguistics (COLING-2010).*,pp.782-790.

[41] Madhyastha, P.S., Carreras, X. and Quattoni, A., 2017. Prepositional Phrase Attachment over Word Embedding Products.*Proceedings of the 15th International Conference on Parsing Technologies.*,pp.32-43.

[42] Maynard, D. and Funk, A.,2011. Automatic Detection of Political Opinions in Tweets. *ESWC 2011 Workshops.*,pp.88-99.

[43] Morinaga, S., Yamanishi, K., Tateishi, K. and Fukushima, T.,2002. Mining Product Reputations on the Web. *In proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD-2002).*

[44] Nasukawa, T. and Yi, J.,2003. Sentiment analysis: Capturing favorability using natural language processing. *In proceedings of the K-CAP-03, 2nd International Conference on Knowledge Capture.*

[45] O'Connor, B., Balasubramanyan, R., Routledge, B. R., and Smith, N.A.,2010. From Tweets to Polls: Linking Text Sentiment to Public Opinion Time Series. *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media.*,pp.122-129.

[46] Pang, B. and Lee, L. and Vaithyanathan, S.,2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. *In proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-2002).*

[47] Pang, B. and Lee, L.,2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval.*,2,pp.1-135.

[48] Pasca, M. and Harabagiou, S.,2001. The Informative Role of WordNet in Open-Domain Question Answering *In proceedings of NAACL 2001 WordNet and other Lexical Resources Workshop.*,pp.138-143.

[49] Rabiner, R.,1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *In proceedings of IEEE,1989*,77(2),pp.257-286.

[50] Ratnaparkhi, A., Reynar, J. and Roukos, S.,1994. A Maximum Entropy Model for Prepositional Phrase Attachment. *In proceedings of the workoshop on Human Language Technology.*,pp.250-255.

[51] Rösiger, I.,2018. Rule-and Learning-based Methods for Bridging Resolution in the ARRAU Corpus.*Proceedings of the Workshop on Computational Models of Reference, Anaphora and Coreference.*,pp.23-33.

[52] Seerat, B. and Azam, F.,2012. Opinion Mining : Issues and Challenges (A survey). *International Journal of Computer Applications.* ,49(9),pp.42-51.

[53] Singh, R. P. and Yavad, P.,2012. Intelligent Information Retrieval in Data Mining. *International Journal of Scientific & Engineering Research.*3,pp.1-6.

[54] Srividhya, V. and Anitha, R.,2010. Evaluating PreprocessingTechniques in Text Categorization *International Journal of Computer Science and Application Issue 2010* pp.49-51.

[55] Stieliz, S. and Xuan, L. D.,2012. Political Communication and Influence through Microblogging-An Empirical Analysis of Sentiment in Twitter Messages and Retweet Behavior.*45th Hawaii International Conference on System Sciences.*,pp.3500-3509.

[56] Takenobu , T., Atsushi, F., Makoto, I., Naoyuki, S. and Hozumi, T., 1997. Extending a thesaurus by classifying words.,pp.16-21.

[57] Tang, H., Tan, S. and Xueqi, C.,2009. A survey on sentiment detection of reviews.*Expert Syst. Appl 36*,7,pp.10760-10773.

[58] Tsytsarsu, M. and Palpanas, T.,2011. Survey on Mining Subjective Data on the Web.*Data Mining and Knowledge Discovery.*,10,pp.1-37.

[59] Turney, P.,2002. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*,pp.417-424.

[60] Vijayarani, S., Ilamathi, J. and Nithya,2015. Preprocessing Techniques for Text Mining - An Overview.*International Journal of Computer Science & Communication Networks.*,pp.7-16.

[61] Vossen, P.,1998. EuroWordNet: A Multilingual Database with Lexical Semantic Networks. *Kluwer Academic Publishers*25(4),pp.628-630.

[62] Wang, H., Can, D., Kazemzadeh, A., Bar, F. and Narayanan, S.,2012. A System for Real-time Twitter Sentiment Analysis of 2012 U.S. Presidential Election Cycle.*Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*,pp.115-120.

[63] Wiebe, J.,1990. Identifying Subjective Characters in Narrative. *In proceedings of the International Conference of Computational Linguistics (COLING-1990)*,pp.401-406.

[64] Wiebe, J.,1994. Tracking Point of Viewin Narrative. *Computational Linguistics)*,pp.233-287. bibitemWiebe, J., Bruce, R., and O' Hara, T.,1999. Development and Use of a Gold-Standard Data Set for Subjectivity Classifications. *In proceedings of the Association for Computational Linguistics(ACL-1999)*,pp.246-253.

[65] Wiebe, J. and Rillof, E.,2005. Creating Subjective and Objective Sentence Classifiers from Unannotated Texts. *Computational Linguistics and Intelligent Text Processing*,pp.486-497.

[66] Yang, Y. and Pedersen, J.,1999. Guest Editors' Introduction: Intelligent Information Retrieval.*In IEEE Intelligent System.*,14[4], pp. 30-31.

[67] Yi, J., Nasukawa, T., Bunescu, R. and Niblack, W.,2003. Sentiment Analyzer: Extracting Sentiments about a Given Topic using Natural Language Processing Techniques.*In Proceedings of International Conference on Data Mining(ICDM-2003).*

[68] Yu, J., Zha, Z., Wang, M. and Chua, T.,2011. Aspect Ranking: Identifying Important Product Aspects from Online Consumer Reviews.*In Proceedings of Annual Meeting of the Associations for Computational Linguistics(ACL-2011).*,pp.1496-1505.

[69] Yu, J., Zha, Z., Wang, M. and Chua, T.,2011. Domain-Assisted Product Aspect Hierarchy Generation: TowardsHierarchical Organization of Unstructured Consumer Reviews*In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing(EMNLP-2011)* .,pp.140-150.

[70] Zhou, X. and Yang, Z.,2013. Sentiment analysis on tweets for social events.*Proceedings of the 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design.*,pp.557-562.

[71] Zhuang, L., Jing, F. and Zhu, X.,2006. Movie Review Mining and Summarization.*Paper presented at the 15th ACM International Conference on Information and Knowledge Management.*,pp.43-50.

[72] Jarmasz, M.,2003. Roget's thesaurus as a lexical resource for natural language processing,Master of Computer Science,School of Information Technology and Engineering,University of Ottawa.

[73] Anon(2019), Alphabetical list of part-of-speech tags used in the Penn Treebank Project. [online] Available at : https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank _pos.html [Accessed 22 Oct. 2019].

[74] Boecher, M.(2019), TFIDF, hyScore.io, Natural language processing solutions. [online] Available at: https://www.hyscore.io/tfidf-vectorization/tfidf [Accessed 22 Oct. 2019].

[75] Mentafloss.com (2019), 10 Fascinating Facts About the Thesaurus for National Thesaurus Day. [online] Available at: https://mentalfloss.com/article/90809/10-fascinating-facts-about-thesaurus [Accessed 22 Oct. 2019].

[76] Snap.stanford.edu. (2019). Index of /data/amazon/productGraph/ categoryFiles. [online] Available at: http://snap.stanford.edu/data/ amazon/productGraph/categoryFiles [Accessed 22 Oct. 2019].

Appendix A

Experiments of domain B)

Experiment's Name	**User's Query**	**Reviews**
B_1	"Cream for sensitive skin!"	*total_B*
B_2	"Powder for sensitive skin!"	*total_B*
B_3	"Cream and powder for sensitive skin!"	*total_B*
B_4	"Cream for rash!"	*total_B*
B_5	"Powder for rash!"	*total_B*
B_6	"Cream and powder for rash!"	*total_B*
B_7	"Cream and powder for rash and generally for sensitive skin!"	*total_B*
B_8	"Cream, powder, and lotion for rash and generally for sensitive skin!"	*total_B*
B_9	"Shampoo and after bath lotion."	*total_B*
B_{10}	"Diapers with long dry."	*total_B*
B_{11}	"Diapers for nightime."	*total_B*
B_{12}	"Eco-friendly diapers!"	*total_B*
B_{13}	" Diapers with long dry for nighttime."	*total_B*

B_{14}	“Bottles for milk.”	*total_B*
B_{15}	“Toys with music!”	*total_B*
B_{16}	“Cream for sensitive skin!”	*part_B_1*
B_{17}	“Cream for sensitive skin!”	*part_B_2*
B_{18}	“Cream for rash!”	*part_B_a*
B_{19}	“Cream for rash!”	*part_B_b*
B_{20}	“Cream for rash!”	*part_B_c*
B_{21}	“Cream for rash!”	*part_B_d*
B_{22}	“Cream for rash!”	*part_B_e*

Table A.1: Experiments of the domain B).

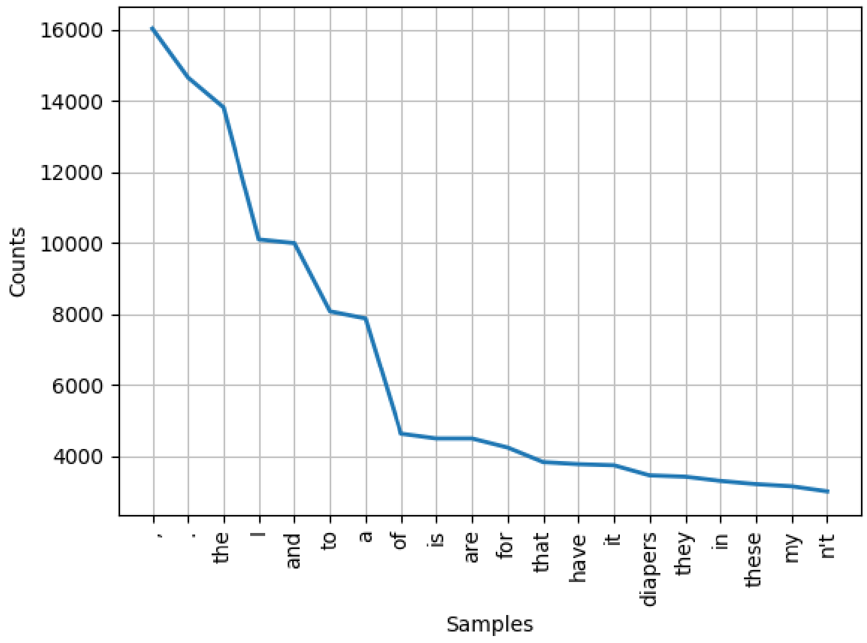
Appendix B

Results of domain B)

Experiment	Reviews	Top_20 words before preprocessing (b.p)	Top_20 words after preprocessing (a.p)	Number of words of the union of reviews (b.p) and (a.p)
$B_1 - B_{15}$	*total_B*	[’,’, ’.’, ’the’, ’I’, ’and’, ’to’, ’a’, ’of’, ’is’, ’are’, ’for’, ’that’, ’have’, ’it’, ’diapers’, ’they’, ’in’, ’these’, ’my’, ”n’t”]	[’diaper’, ’nt’, ’diaper’, ’baby’, ’pamper’, ’like’, ’use’, ’wipe’, ’leak’, ’get’, ’size’, ’brand’, u’try’, ’great’, ’huggies’, ’well’, ’work’, ’little’, ’ve’, ’time’]	(b.p)=393.696 (a.p)=171.268
B_{16}	*part_B_1*	[’,’, ’.’, ’the’, ’I’, ’and’, ’to’, ’a’, ’of’, ’is’, ’it’, ’for’, ’are’, ’have’, ’that’, ’diapers’, ’in’, ’my’, ’they’, ’these’, ’with’]	[’diaper’, ’nt’, ’diaper’, ’baby’, ’pamper’, ’like’, ’leak’, u’use’, ’get’, ’huggies’, ’size’, ’try’, ’little’, ’great’, ’fit’, ’well’, ’brand’, ’work’, ’love’, ’dry’]	(b.p)=185.235 (a.p)=80.800
B_{17}	*part_B_2*	[’,’, ’.’, ’the’, ’and’, ’I’, ’to’, ’a’, ’are’, ’of’, ’is’, ’for’, ’that’, ’have’, ’they’, ’diapers’, ’it’, ’these’, ’in’, ”n’t”, ’my’]	[’diaper’, ’nt’, ’diaper’, ’wipe’, ’baby’, ’like’, ’pamper’, ’use’, ’leak’, ’get’, ’brand’, ’well’, ’great’, ’try’, ’work’, ’size’, ’ve’, ’time’, ’little’, ’good’]	(b.p)=208.460 (a.p)=90.468
B_{18}	*part_B_a*	[’.’, ’,’, ’the’, ’I’, ’and’, ’to’, ’a’, ’it’, ’is’, ’for’, ’of’, ’that’, ’my’, ’have’, ’are’, ’in’, ’on’, ”n’t”, ’but’, ’they’]	[’nt’, ’baby’, ’diaper’, ’diaper’, ’pamper’, ’like’, ’use’, ’get’, ’leak’, ’huggies’, ’try’, ’size’, ’work’, ’love’, ’buy’, ’great’, ’little’, ’son’, ’brand’, ’well’]	(b.p)=70.693 (a.p)=30.453

B_{19}	*part_B_b*	[’,’, ’.’, ’the’, ’I’, ’and’, ’to’, ’a’, ’of’, ’are’, ’is’, ’diapers’, ’for’, ’have’, ’that’, ’it’, ’in’, ’these’, ’they’, ’my’, ’with’]	[’diaper’, ’diaper’, ’nt’, ’baby’, ’like’, ’huggies’, ’leak’, ’use’, ’wipe’, ’get’, ’pamper’, ’little’, ’fit’, ’size’, ’great’, ’well’, ’brand’, ’try’, ’work’, ’also’]	(b.p)=69.800 (a.p)=30.587
B_{20}	*part_B_c*	[’,’, ’.’, ’the’, ’I’, ’and’, ’to’, ’a’, ’of’, ’are’, ’is’, ’diapers’, ’for’, ’they’, ’that’, ’these’, ’have’, ’in’, ”n’t”, ’with’, ’it’]	[’diaper’, ’nt’, ’diaper’, ’pamper’, ’like’, ’baby’, ’use’, ’leak’, ’get’, ’size’, ’brand’, ’well’, ’fit’, ’try’, ’great’, ’dry’, ’little’, ’ve’, ’work’, ’seem’]	(b.p)=89.850 (a.p)=39.294
B_{21}	*part_B_d*	[’,’, ’.’, ’the’, ’and’, ’I’, ’to’, ’a’, ’is’, ’of’, ’are’, ’for’, ’it’, ’that’, ’have’, ’my’, ’these’, ’on’, ’they’, ’in’, ”n’t”]	[’wipe’, ’nt’, ’diaper’, ’diaper’, ’baby’, ’like’, ’use’, ’get’, ’pamper’, ’great’, ’leak’, ’work’, ’try’, ’well’, ’time’, ’huggies’, ’ve’, ’night’, ’make’, ’good’]	(b.p)=72.081 (a.p)=31.261
B_{22}	*part_B_e*	[’,’, ’.’, ’the’, ’and’, ’I’, ’a’, ’to’, ’are’, ’of’, ’is’, ’have’, ’for’, ’that’, ’they’, ’diapers’, ’in’, ”n’t”, ’it’, ’these’, ’with’]	[’diaper’, ’nt’, ’diaper’, ’pamper’, ’wipe’, ’baby’, ’like’, ’leak’, ’use’, ’brand’, ’get’, ’size’, ’ve’, ’try’, ’well’, ’great’, ’seem’, ’time’, ’work’, ’night’]	(b.p)=91.270 (a.p)=39.673

Table B.1: Results of the pre-processing of data of the domain B).



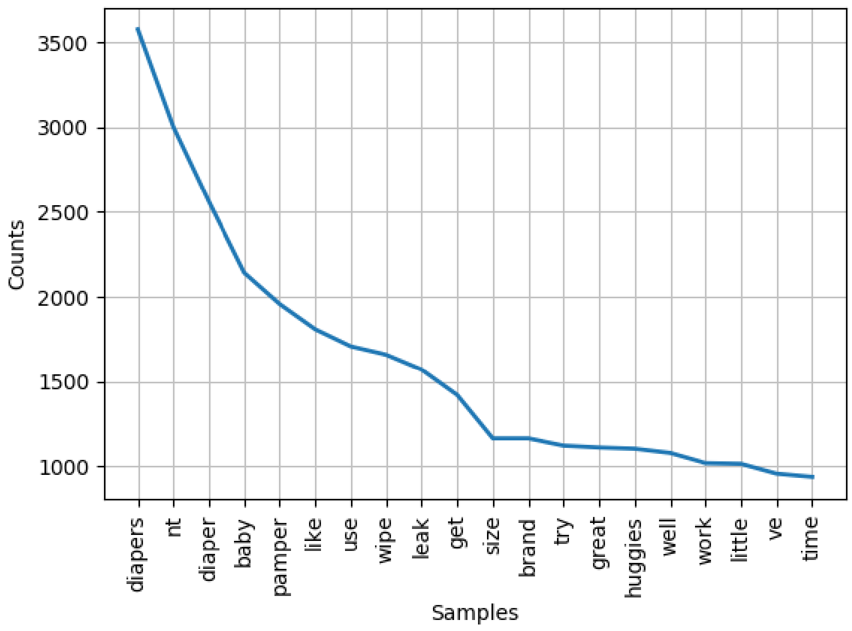


Figure B.1: Twenty most frequent words for the first 15 experiments of domain B) before(up) an after(down) pre-processing.

Experiment	**Reviews**	**Top-aspects (TOP_NN_VB_JJ)**	**Thesauri of top aspects**
$B_1 - B_{15}$	*total_B*	['diaper', 'baby', 'great', 'size', 'little', 'good', 'time', 'night', 'dry', 'son', 'brand', 'skin', 'fit', 'old', 'daughter', 'sensitive', 'product', 'price', soft', 'work', 'love', 'rash', 'smell', 'change', 'easy', 'leak', 'nice', 'new', 'big', 'free', 'regular',...]	Example of aspect : diaper **synonyms_of_diaper**=['diaper', 'nappy', 'napkin''] **hypernyms_of_diaper**=['garment', 'fabric'] **hyponyms_of_diaper**=[ ]
B_{16}	*part_B_1*	['diaper', 'baby', 'size', 'great', 'little', 'good', 'dry', 'son', 'time', 'night', 'fit', 'daughter', 'rash', 'old', 'price', 'brand', 'skin', 'product', 'sensitive', 'soft', 'day', 'love', 'work', 'buy', 'smell', 'easy', 'leak', 'new', 'big', 'nice', 'sure',...]	Example of aspect : baby **synonyms_of_baby**=['babe', 'infant', 'child', 'sister', 'pamper',] **hypernyms_of_baby**=['child', 'treat',...] **hyponyms_of_baby**=['cherub', 'neonate', 'nursling', 'papoose',...]
B_{17},	*part_B_2*	['diaper', 'baby', 'great', 'size', 'good', 'little', 'night', 'time', 'brand', 'son', 'generation', 'skin', 'dry', 'sensitive', 'old', 'fit', 'daughter', 'price', 'work', 'love', 'day', 'feel', 'quality', 'soft', 'nice', 'overnight', 'free', 'easy', 'regular', 'leak',...]	Example of aspect : night **synonyms_of_night**=['night', 'nighttime', 'dark'] **hypernyms_of_night**=['dark', 'twilight'] **hyponyms_of_night**= ['wedding_night', 'weeknight']
B_{18}	*part_B_a*	['baby', 'diaper', 'great', 'size', 'little', 'son', 'good', 'dry', 'time', 'daughter', 'rash', 'night', 'product', 'brand', 'old', 'price', 'easy', 'skin', 'fit', 'love', 'work', 'buy', 'cream', 'smell', 'day', 'soft', 'big', 'leak', 'sensitive', 'sure', 'small', 'newborn', 'nice,...']	Example of aspect : rash **synonyms_of_rash**=[rash', 'roseola', 'efflorescence', 'blizzard','reckless',...] **hypernyms_of_rash**=['eruption', 'series'] **hyponyms_of_rash**= ['prickly_heat', 'urtication']

B_{19}	*part_B_b*	[’diaper’, ’baby’, ’little’, ’great’, ’size’, ’good’, ’time’, ’fit’, ’night’, ’price’, ’brand’, ’skin’, ’son’, ’day’, ’product’, ’soft’, ’old’, ’sensitive’, ’dry’, ’love’, ’work’, ’rash’, ’change’, ’smell’, ’nice’, ’easy’, ’natural’, ’leak’, ’new’, ’happy’,...]	Example of aspect : skin **synonyms_of_skin**=[’skin’, ’tegument’, ’pelt’, ’peel’, ’scramble’, ’shin’, u’shinny’,’scrape’, ’pare’,...] **hypernyms_of_skin**=[’surface’, ’rind’,...’]] **hyponyms_of_skin**=[’cuticle’, ’dewlap’, ’hangnail’, ’prepuce’, ’scalp’,...]
B_{20}	*part_B_c*	[’diaper’, ’size’, ’baby’, ’great’, ’little’, ’dry’, ’good’, ’generation’, ’fit’, ’old’, ’sensitive’, ’son’, ’brand’, ’rash’, ’skin’, ’night’, ’day’, ’work’, ’love’, ’price’, ’free’, ’soft’, ’leak’, ’different’, ’easy’, ’comfortable’, ’disposable’,...]	Example of aspect : comfortable **synonyms_of_comfortable**=[’easy’, ’comfy’, ’prosperous’,...] **hypernyms_of_comfortable**=[ ] **hyponyms_of_comfortable**=[ ]
B_{21}	*part_B_d*	[’diaper’, ’baby’, ’great’, ’night’, ’good’, ’little’, ’time’, ’skin’, ’size’, ’son’, ’sensitive’, ’daughter’, ’old’, ’product’, ’price’, ’overnight’, ’easy’, ’love’, ’monitor’, ’work’, ’brand’, ’day’, ’camera’, ’soft’, ’dry’, ’regular’, ’nice’, ’free’, ’big’, ’clean’,...]	Example of aspect : soft **synonyms_of_soft**=[’soft’, ’delicate’, ’indulgent’, ’easy’, ’flabby’, ’flaccid’, ’cushy’, ’balmy’,...] **hypernyms_of_soft**=[ ] **hyponyms_of_soft**=[ ]
B_{22}	*part_B_e*	[’diaper’, ’baby’, ’size’, ’great’, ’night’, ’little’, ’time’, ’dry’, ’good’, ’brand’, ’son’, ’skin’, ’fit’, ’old’, ’product’, ’sensitive’, ’soft’, ’price’, ’wipe’, ’work’, ’box’, ’love’, ’quality’, ’daughter’, ’day’, ’change’, ’feel’, ’leak’, ’nice’, ’new’, ’extra’, ’overnight’, ’regular’, ’clean’,...]	Example of aspect : wipe **synonyms_of_wipe**=[’rub’, ’wipe’, ’pass_over’] **hypernyms_of_wipe**=[’rub’, ’contact”] **hyponyms_of_wipe**=[scuff’, ’sponge’, ’squeegee’, ’sweep’, ’towel’, ’whisk’]

Table B.2: Results of the aspects’ extraction and expansion of them on thesauri of the domain B).