Mining Comparative Opinions using Multi-label Machine Learning Techniques

A case study to identify comparative opinions, based on product aspects, and their sentiment classification, in online customer reviews.

PAPER WITHIN *Software Product Engineering Master's Program*
AUTHOR: *Yassin Haj Ahmad*
TUTOR: *Johannes Schmidt*
JÖNKÖPING December 2018

This exam work has been carried out at the School of Engineering in Jönköping in the subject area "Mining Comparative Opinions using Multi-label Machine Learning Techniques". The work is a part of the two-years Master of Science programme, Software Product Engineering.

The author takes full responsibility for opinions, conclusions and findings presented.

Examiner: Ulf Johansson

Supervisor: Johannes Schmidt

Scope: 30 credits (second cycle)

Date: 2018-12-10

Praise be to Allah.

Thanks to my parents, brothers, sister and my wife Sara, for their continuous love and support.

My appreciation to the supervisor, examiner and course manager of the thesis for their help and advice that contributed significantly to this work.

Abstract

There is a high demand to summarize and analyze the opinions in online customer reviews. Sentiment analysis is one of the study fields in this area. Mining comparative opinions is an important application of sentiment analysis. It includes identifying the comparative opinions and the aspects that are compared. It also identifies the sentiment classification of the opinion as positive or negative. This helps businesses to make effective decisions in the development and promotion of their products and services, and to better understand their competitors. Different approaches could be used to address this sentiment analysis application, such as Machine Learning. The application is a multi-label classification problem from a machine learning perspective. This paper presents a case study to evaluate three multi-label machine learning classification techniques in addressing the problem. Empirical experiments are conducted on a domain-independent dataset of online customer reviews from Amazon for the evaluation purpose.

Summary

Identifying comparative opinions based on product aspects and their sentiment classification is an important sentiment analysis application. Different approaches and techniques have been proposed to address this problem. With the continuous development of machine learning, new techniques have been introduced such as multi-label machine learning techniques. The purpose of this research is an evaluation of using these techniques in addressing the problem of this sentiment analysis application.

A dataset of domain-independent online customer reviews from Amazon was used to develop the classification model for the evaluation of multi-label classification techniques. The reviews in the dataset were labeled manually by humans for the comparative and aspects-based labels. The labeled dataset was preprocessed and used to develop a classification model using Python and Scikit-learn tool.

Three machine learning multi-label classification techniques were evaluated; Problem Transformation, Algorithm Adaptation and Ensemble (RAKEL). The techniques were evaluated in terms of their effectiveness and efficiency in addressing the sentiment analysis application. The efficiency of the techniques was determined based on the performance of different machine learning algorithms. Then, the effectiveness was determined based on the efficiency.

The evaluated multi-label techniques were found to be effective in addressing the problem in the application. The most efficient technique is the Problem Transformation using Binary Relevance method with Maximum Entropy algorithm which scored a Macro F1-score of 85.3%.

Keywords

Sentiment Analysis, Opinion Mining, Machine Learning, Multi-label, Text Classification, Comparative, Aspect-based.

Contents

List of Tables

List of Figures

List of Abbreviations[1]

NLP	Natural Language Processing
NB	Naïve Bayes
NBML	Naïve Bayes Multi-label
SVM	Support Vector Machine
MaxEnt	Maximum Entropy
kNN	k-Nearest Neighbors
DT	Decision Trees
ML-DT	Multi-label Decision Trees
RF	Random Forest
MLkNN	Multi-label k-Nearest Neighbors
RAKEL	Random k-Labelsets
BoW	Bag of Words
NN	Neural Networks
OVA	One-vs-All
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
DSRP	Design Science Research Process
NLTK	Natural Language Tool Kit
MP	Macro Precision
MR	Macro Recall
MF1	Macro F1-score
HL	Hamming Loss
AC	Accuracy

[1] The abbreviations are listed according to their order in the paper.

1 Introduction

With the huge expansion of the web and the flourishing number of providers of online services and products, there is a massive amount of data growing rapidly. Part of this data is the textual information written by people in different forms such as comments and reviews. Websites like Amazon and Trip Advisor encourage customers to write reviews about their services and products eagerly (Abramyk, 2017; Hu & Liu, 2004). This helps them in effective decisions making to improve their business. Also, it provides other people with useful information when thinking to buy a product. With more and more people using these websites, the number of reviews is growing very fast. This makes difficult for the providers to keep track of them manually. It also becomes difficult for customers to read all the reviews to find relevant information (Horrigan, 2008).

Therefore, the demand is increasing to understand and analyze the opinions in online customer reviews using systematic techniques. Sentiment Analysis, also known as Opinion Mining, is one of the common practices used for this purpose. It can be simply defined as the analysis of opinions, attitudes and emotions in text (Varghese & Jayasree, 2013). Sentiment analysis covers many applications such as sentiment classification, subjectivity analysis, mining comparative opinions, viewpoints identification, emotions-based classification and product aspects extraction (Pang & Lee, 2008). These applications have been explored in many researches and used in many business areas (Mäntylä, Graziotin, & Kuutila, 2018).

For instance, mining comparative opinions in online customer reviews can provide very useful insights for businesses to maintain their competitive edge (Xu, Liao, Li, & Song, 2011). Identifying reviews that compare a product with another one based on specific aspects has a great business value towards improving the product (Jian, Ji, & Yan, 2017). This can be achieved by investigating the aspects that got negative feedbacks for possible improvements. Also, identifying the aspects that received positive feedbacks can be helpful for marketing and promotions. Therefore, it is considered an important field of study where new techniques need to be used and evaluated continuously.

Most literature discuss two main approaches in sentiment analysis and opinion mining applications. First, the lexicon-based approach in which techniques from statistics, linguistics, natural language processing (NLP)[2], and semantic dictionaries are used to identify the sentiment of an opinion (Turney, 2002; Vaghela & Jadav, 2016). Second, the machine learning approach. In this approach different machine learning algorithms are used to explore and learn the sentiments of opinions in a set of available data. Then, using the learning outcome to analyze the opinions in a new set of data (Pang, Lee, & Vaithyanathan, 2002).

Machine learning approach is believed to outperform other approaches in sentiment analysis (Pang, Lee, & Vaithyanathan, 2002). Algorithms such as Naïve Bayes (NB), Maximum Entropy (MaxEnt), and Support Vector Machines (SVM) proved successful sentiment classification results (Pang, Lee, & Vaithyanathan, 2002).

[2] Abbreviation is done for terminology of three or more words and are used frequently in this thesis.

1.1 Background

The demand is increasing to understand opinions in online customer reviews using systematic techniques (Hu & Liu, 2004). Reviews may contain a direct opinion by the customer as a positive or negative feedback. However, there are often more information in the customer opinion. For example, the opinion can be expressed on a specific aspect or entity of a product[3]. It may also contain a comparison with another product or an aspect of that product. These reviews are more useful for the product providers (Murthy & Liu, 2008). Identifying these reviews is done by combining three different sentiment analysis applications. These applications are mining comparative opinions, identifying product aspects and sentiment classification (Aggarwal, Zhai, & editors, 2012).

Practitioners have widely studied these sentiment analysis applications in separate researches (Varathan, Giachanou, & Crestani, 2017; Medhat, Hassan, & Korashy, 2014). New techniques and improvements are proposed continuously using different approaches such as statistics, NLP and machine learning. Surveys and comparative studies are performed as well to evaluate the proposed approaches and techniques (Ghag & Shah, 2013; Varghese & Jayasree, 2013). For example, comparing lexicon-based with machine learning approaches for specific sentiment analysis application (Turney, 2002; Cui, Mittal, & Datar, 2007). This indicates that sentiment analysis is an important field of study.

One of the earliest researches that combined comparative opinion mining with aspects identification is the work done by Murthy & Liu (2008). The application is identifying which entity in a comparative opinion is the preferred one. The authors used a heuristic algorithm to identify this preferred entity based on patterns and relations extraction. Furthermore, two of the newest researches in this field are published by Wang, et al. (2017) and Wang et al. (2018). They used statistical modeling techniques such as Conditional Random Fields and Latent Dirichlet Allocation to extract comparative information from text. In a research published by Tkachenko & Lauw (2014), a machine learning approach was used to build a statistical model to understand the comparison between entities and find the entity of interest. It can be concluded from the researches above that using machine learning techniques as a single solution needs more evaluation.

When using the machine learning approach in sentiment analysis, classification techniques are most commonly used (Medhat, Hassan, & Korashy, 2014). Machine learning classification techniques are mostly used for binary or multi-class classification problems. Binary classification means assigning a class for each data record among two available classes, while multi-class means that there are more than two classes to be assigned. In both cases, each record has only one label (Tan, 2006). With the continuous development of machine learning, advanced techniques have been proposed to handle multi-label classification problems (Zhang & Zhou, 2014). Multi-label classification means that there are several labels with two or more classes available to be assigned to each data record (Tsoumakas & Katakis, 2007).

[3] For simplicity, the term "aspects" is used in this document to indicate all the aspects, features, properties, attributes and entities of the product.

Multi-label machine learning classification techniques can be used to solve sentiment analysis problems when the application requires assigning multiple labels. Mining comparative opinions based on product aspects is one of the sentiment analysis applications that is considered a multi-label classification problem. There are three labels to be identified and assigned to each review (Sentiment, Comparative, Aspects-based). Similarly, many other sentiment analysis applications are considered multi-label problems (Lima & Castro, 2014; Wei, Zhang, Zhang, Li, & Miao, 2011; Liu & Chen, 2015). Therefore, evaluating multi-label machine learning techniques is a good contribution to this field of study.

There are three main multi-label classification techniques (Zhang & Zhou, 2014; Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012). They can be summarized as follows:

1. Problem Transformation: the concept of this technique is transforming the multi-label classification problem into multiple single-label problems of binary or multi-class classification. It includes multiple methods such as Binary Relevance, Label Powerset and Classifier Chain. Accordingly, there are several machine learning algorithms that can be used in these methods.
2. Algorithm Adaptation: this technique refers to adapting machine learning algorithms to handle multi-label classification problems, such as adapting k-Nearest Neighbors (kNN) algorithm to MLkNN (Zhang & Zhou, 2007).
3. Ensemble: both of the techniques above are used by this technique to construct an ensemble of optimized algorithms to perform multi-label classification. A common method of the ensemble technique is Random k-Labelsets (RAKEL).

Many surveys about sentiment analysis and comparative opinion mining have been published recently (Varathan, Giachanou, & Crestani, 2017; Mäntylä, Graziotin, & Kuutila, 2018; Varghese & Jayasree, 2013). It can be observed in these surveys that the use of multi-label machine learning techniques above has not been studied to address this type of sentiment applications. This creates an opportunity for further research and investigation in this area.

1.2 Purpose and Research Questions

This thesis studies the sentiment analysis application of identifying comparative opinions based on product aspects, and their sentiment classification, in online customer reviews. This application is considered a multi-label classification problem from machine learning perspective. The purpose of this study is to evaluate the use of different multi-label machine learning classification techniques in this sentiment analysis application.

The multi-label classification techniques[4] are Problem Transformation, Algorithm Adaptation and Ensemble. These techniques involve different methods and can be applied to several machine learning algorithms. Therefore, empirical experiments

[4] For simplicity, the term "multi-label classification technique" is used without adding machine learning to it since the general approach in this research is machine learning.

are needed to evaluate the effectiveness and efficiency of each technique. This is done by measuring the performance of different machine learning algorithms when they are used in the technique.

The performance of the algorithm is measured by a set of common metrics used to evaluate the machine learning classification models such as precision, recall, F1-score and accuracy (Section 2.7). It also includes the time measurement. The efficiency of the technique is evaluated by the best performance of the algorithms when it is used in the technique. The effectiveness of the technique is decided by evaluating its efficiency. Therefore, the effectiveness is determined by asking whether the technique is good or not. The efficiency is determined by asking how good the technique is.

The main research question in this thesis is "How good is using multi-label classification techniques in identifying comparative opinions based on product aspects and their sentiment classification?".

By taking into consideration that each technique is applied using several algorithms, and the context is the sentiment analysis application in this research. Then, the sub-questions are:

1. What is the performance of machine learning algorithms when they are used in each multi-label classification technique?
2. What is the efficiency of each multi-label classification technique in terms of the best performance of the machine learning algorithms?
3. What is the effectiveness of each multi-label classification technique?

The contribution of this thesis is the evaluation of a multi-label machine learning solution for the sentiment analysis application under study. This is important from three perspectives:

1. Sentiment analysis perspective: it helps in finding an effective multi-label machine learning solution to the sentiment analysis application.
2. Machine learning perspective: it gives a good idea about the efficiency of different multi-label classification techniques in sentiment analysis in general.
3. Business perspective: the sentiment analysis application has a great business value for companies that are looking to implement such an application on the online customer reviews of their products and services.

1.3 Related Work

Mining comparative opinions, identifying product aspects and sentiment classification applications have been studied widely in researches. They are usually studied as separate applications (Appel, Chiclana, & Carter, 2015). In this thesis, these applications are studied as a single application with a multi-label machine learning approach. Below are some of the work related to the research in this thesis.

Xu et al. (2011) conducted an experiment to extract and visualize comparative relations between products in Amazon online customer reviews. The authors claim

that their method achieved an average accuracy of 63%. They used advanced linguistics features to identify comparison relations, which includes both comparison identification and direction of the comparison. However, the work does not involve machine learning.

Jindal & Liu (2006) investigated the identification of comparative opinions and showed multiple factors that affect the identification such as linguistics. They used their findings in another research to identify the preferred entity in the comparative opinions (Murthy & Liu, 2008). However, statistical approaches were mainly used.

Mubarok et al. (2017) investigated aspect-based sentiment analysis on inline product reviews using Naïve Bayes. Their proposed method achieved F1-Measure of 78.12%. This paper is related to the aspects-based classification part done in this thesis. However, mining comparative opinions was not studied.

Khan et al. (2016) used Naïve Bayes algorithm in multi-label classification of a comments on specific video. In their application, they did a comparative sentiment analysis between Android and iPhone. They have two labels, which are the models, then three classes for each as Positive, Negative and Neutral. They assumed that the words around the keyword are enough to understand the sentiment. This is in order to reduce complexity and increase the performance of the algorithm. They obtained an approximate accuracy of 60%. The nature of the research is different as it is a domain dependent application.

Liu & Chen (2015) proposed a multi-label machine learning technique for extracting emotions from microblogs. They used microblogs of two major incidents in China to classify the sentiment of emotions in them. Their approach consists of three components; Text segmentation, feature extraction and multi-label classification. Programmable techniques have been used for text segmentation. Then, they used three different dictionaries for features extraction. At the end, they compared the performance of 11 different machine learning algorithms. Their work is related to this research in terms of using multi-label techniques and extracting aspects from text. However, it does not include comparative opinion analysis. The authors claim that it is the first multi-label work in the sentiment analysis application studied in the paper.

Wei et al. (2011) used a Naïve Bayesian Multi-label (NBML) classification algorithm to visualize text search results. The classification in NBML is done by transforming the multi-label classification problem into the combination of several single-label NB classifiers. They relied on an adapted strategy for features selection to reduce the computational complexity. Their experiment showed that the algorithm has a competitive performance over other algorithms.

Jian et al. (2017) used structured reviews that have pros and cons to identify comparison between products for competitive intelligence. They used clustering as the main machine learning technique for this purpose. Also, they have used structured reviews that have pros and cons in them, which makes it different than the work done in this thesis.

1.4 Assumptions

The assumptions in this case study are summarized as follows:

1. Domain independent: the selection of the online customer reviews dataset, the design of the research method, and the evaluation presented in this research is independent of the domain, provider, reseller or product.

2. Labeling of the reviews in the dataset is done manually and with help of three graduate students, who represent possible customers and users. This is assumed to be easy and reliable. Thus, the findings of this thesis are based on that assumption.

3. No dependency between labels: it is assumed in this case study that the labels are totally independent. This is taken in consideration when implementing the multi-label classification techniques in this thesis. However, it should be mentioned that minor relations were observed while labeling the dataset. Negative opinions are more likely to have comparison with other product to justify the negative review. Also, comparative opinions are more likely to be on product aspects rather than the whole product.

4. The sentiment analysis is done at the document level: sentiment analysis is done on three main levels: document, sentence and aspect-based levels (Medhat, Hassan, & Korashy, 2014). The analysis level depends on the application requirements and its complexity. This thesis implements the sentiment analysis on the document level, which represent in this case an online customer review.

1.5 Delimitations

Due to the scope limitation of the thesis, the following topics were not covered:

1. Further analysis on the comparative opinions and aspect-based identified in the online customer reviews.

2. Comparison with other statistical, NLP or machine learning approaches used for addressing similar sentiment analysis problems.

3. The purpose of the case study is an evaluation of the techniques rather than finding an optimum solution to the sentiment analysis application. Therefore, comparing these techniques is not in the scope of this work.

4. Optimizing each of the algorithms to get the best classification performance. The classification is mostly done with default parameters for each algorithm when it gives a fair performance. Some optimization is done implicitly during the evaluation, but the process is not described for the purpose of keeping the work focused on a higher-level evaluation.

5. Reviews with a neutral class that has rating 3 in the source dataset were skipped. It was noticed that these reviews are expressed either negatively or positively which was hard to judge and can affect the classification.

1.6 Outline

The sections in this thesis report are organized in this order: Introduction, Theoretical Background, Research Method and Implementation, Findings and Analysis, Discussion and Conclusion, and References.

The background, purpose and research questions are presented in the introduction section. The section also contains the assumptions, delimitations and related work parts.

The research methods which are used in this thesis are presented in the method and implementation section. This section includes the process followed to carry the research work and the implementation results of the work.

In the findings and analysis section, the implementation results are used and analyzed to answer the research main question and sub-questions. This is done by answering each sub-question in each part of the section.

Finally, a discussion and conclusion on the method and findings are presented in the Discussion and Conclusion section.

2 Theoretical Background

2.1 Sentiment Analysis

It can be defined as "the field of study that analyzes people's opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes." (Liu B. , 2012).

Some researchers and practitioners refer to sentiment analysis in the literature as Sentiment Classification or Opinion Analysis, Opinion Extraction, sentiment Mining, and Subjectivity Analysis or Classification (Pang & Lee, 2008). However, they are all considered under the umbrella of sentiment analysis even though they are slightly different (Liu B. , 2012).

Sentiment analysis uses various techniques from Statistics, Natural Language Processing (NLP), and Machine Learning (Varghese & Jayasree, 2013). The practice initially emerged from NLP, which is the concept of using computer systems to manipulate unstructured text to get useful insights (Chowdhury, 2003). Furthermore, the high performance and precision of machine learning algorithms have proven high capabilities in mining text data.

Sentiment analysis is also considered a subfield of Data Mining or more specifically Text Mining, which is the practice of discovering hidden knowledge from large amount of text data (Tan, 2006). This is because sentiment analysis follows a very similar process in terms of data understanding, preprocessing and representation as in data mining.

Sentiment analysis has many applications where different approaches are used. There are also different techniques in each approach. In the following sections, a brief background is given on these topics. Additionally, providing more details on machine learning algorithms and multi-label classification techniques that are used in this thesis.

2.2 Sentiment Analysis Applications

By reviewing different literature and researches, it can be observed that there are several sentiment analysis applications. Four common applications are Sentiment Classification, Subjectivity Analysis, Mining Comparative Opinions Analysis and Aspect-based Classification (Pang & Lee, 2008).

2.2.1 Sentiment Classification

Classification of the opinion sentiment is the main objective of sentiment analysis. It can be defined as identifying the sentiment in any part of a text and classifying it as Positive or Negative (Ye, Zhang, & Law, 2009). However, a third class of Neutral can be added to the classification. Some researchers considered that objective sentences as neutral. However, this is not always true since the opinion in a sentence could be preserved or not given, and it is just a declarative comment by the writer, rather than being neutral. Therefore, applications tend to use a binary classification of two classes. Also, it is not necessary to use the degree of positivity as the

sentiment classes. There are different interpretations of positive / negative such as supports / does not support, like / dislike, thumps up / thumbs down and likely to win / likely to lose (Pang & Lee, 2008).

2.2.2 Subjectivity Analysis

Most of sentiment analysis applications usually involve subjectivity analysis in one way or another. Intuitively, when it is required to classify the sentiment in a document, in requires first to identify these sentences in the document that contain opinions, emotions or attitudes, which are considered subjective sentences. Then, discard irrelevant sentences that have no sentiment or irrelevant. They are called objective sentences (Varghese & Jayasree, 2013).

2.2.3 Aspect-based Identification

In online product reviews, customers usually mention one or more aspects of the product and give their opinion about them. Therefore, it is more beneficial to identify if the opinion is given on the overall product or specific aspects of it (Rao, Murthy, & Adinarayana, 2017). This sentiment analysis application is also called Aspects-based Classification or sometimes called Features or Entity Extraction in the literature (Vohra & Teraiya, 2013).

2.2.4 Mining Comparative Opinions

Many customers tend to give a comparison with other products in their reviews or compare an aspect of the product under review with the same aspect in another product. This is an important part of the review for both the provider and potential customers, since it gives more useful information and authenticity to the feedback given by the reviewer (Khan, Khan, & Khan, 2016). Comparative opinions analysis applied along with aspect-based classification provides deep understanding of the review and more accurate and comprehensive sentiment analysis.

2.3 Sentiment Analysis Process

Data preprocessing is the initial step in any sentiment analysis application. It includes data cleaning and preparation for next steps. Data cleaning involves the removal of stop words, special characters, unwanted punctuation, new lines, ASCII codes, emoticons, irrelevant languages words etc. (Vaghela & Jadav, 2016). Data preparation involves tokenization, expanding contractions, negation handling, and using other tools from the NLP field (Bhadane, Dalal, & Doshi, 2015).

After that, subjectivity analysis is usually done to identify the subjective content in the text data and ignore the objective content. This helps in reducing the complexity of the sentiment analysis and getting better results. At the end, the classification of the sentiment in the text is implemented using different techniques. The process can also involve other steps depending on the requirements of the application, such as identifying comparative opinions and product aspects.

The overall process of the sentiment analysis can be illustrated in Figure 1 below (Vaghela & Jadav, 2016) (Medhat, Hassan, & Korashy, 2014).

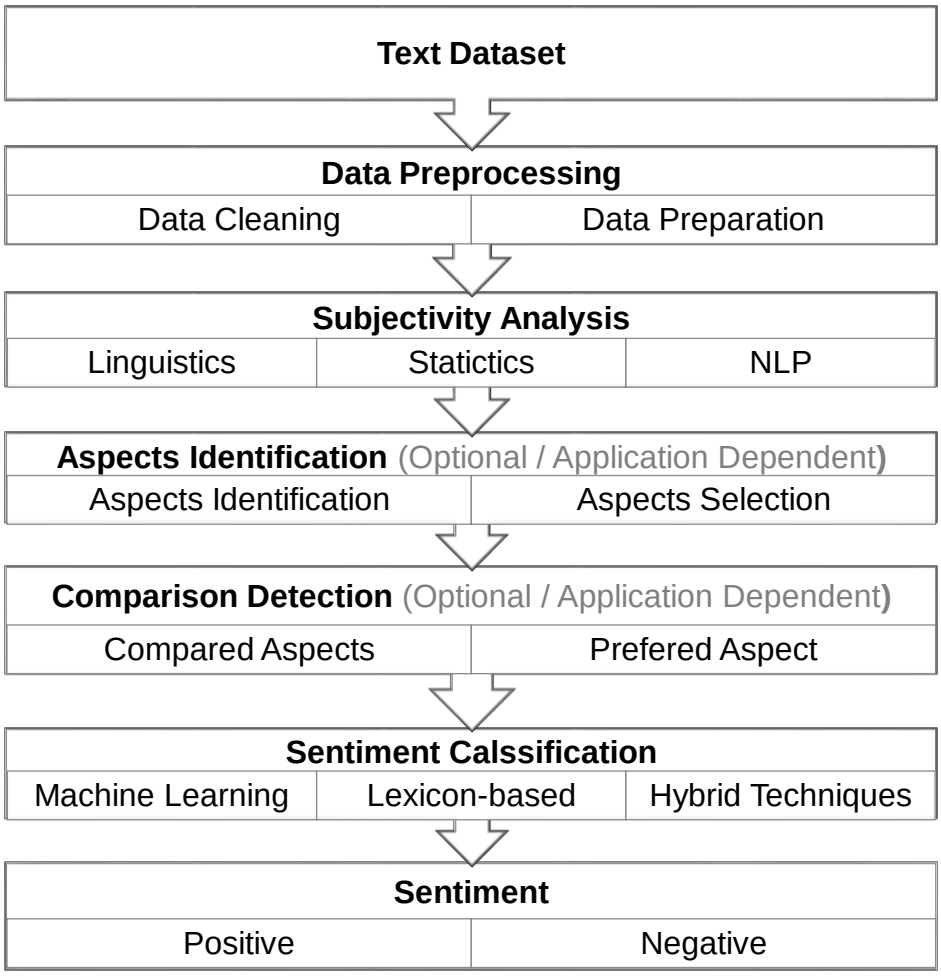

Figure 1. Basic Process of Sentiment Analysis.

2.4 Sentiment Classification

In the sentiment classification step, which is the core of the sentiment analysis, either specific technique from machine learning or lexicon-based approaches is used. A combination of both approaches in a hybrid classification approach can be used as well (Ahmad, Aftab, Ali, & Hameed, 2017).

The selected techniques for classification are taken in consideration when applying sentiment classification. Also, when applying features extraction, comparative analysis if required by the application. In addition to that, data preprocessing step will also be affected based on the classification techniques used (Tan, 2006). Therefore, the proposed research work in sentiment analysis is usually titled and grouped under these three classification categories (Thakkar & Patel, 2015).

Lexicon-based approach: In this category most of the analysis are dictionary-based techniques. The dictionaries, called Lexicons, contain semantic words with their sentiment weights. These words are pre-prepared by the community of researchers and practitioners in the sentiment analysis field such as WordNet and SentiWordNet (Varghese & Jayasree, 2013). The approach is to compare the phrases in a given text with phrases in lexicons to classify the sentiment.

Machine learning approach: The machine learning techniques in general are divided the two sub categories of Supervised and Unsupervised Learning. The use of these techniques depends on the presence or the absence of a training dataset that is labeled already. Most of the sentiment analysis applications adopt supervised learning techniques (Medhat, Hassan, & Korashy, 2014). However, some other researchers suggested the use of clustering with suitable algorithms such as K-means in sentiment analysis, which is considered unsupervised techniques (Unnisa, Ameen, & Raziuddin, 2016).

2.4.1 Lexicon-based Approach

In this approach, a lexicon-based, or sometimes called Lexical, techniques are used to determine the semantic orientation of sentences, phrases and words (Vaghela & Jadav, 2016). Some researches use the term unsupervised learning for this approach (Medhat, Hassan, & Korashy, 2014).

The semantic orientation is determined by calculating how much a part of the text is related to the word "excellent" or the word "poor". The analyzed text is given a class of Positive, Negative or Neutral based on the average semantic orientation of all parts of the text (Turney, 2002). There are several lexicon sources and dictionaries that help in finding the semantic orientation such as WordNet and SentiWordNet. WordNet is an English lexicon of verbs, adjectives, adverbs and nouns with semantic relations. SentiWordNet is built upon WordNet and gives semantic scores to each word. Using these sources, the semantic orientation can be determined by assigning +1 to positive words, +0.5 to weak positive words, -1 to negative words, -0.5 to weak negative words and 0 to neutral words as semantic orientation score (Vaghela & Jadav, 2016).

Using lexical techniques requires a tokenizer at the data preparation step, where the input text is transformed into tokens. Then, a search is done in lexical sources for each token. If found, then the score is summed to the total score of the input text. That means if the word is positive its score will be added to the total, and if negative it will be deducted (Thakkar & Patel, 2015). Researches have proposed so many methods and techniques to adapt this basic approach into more powerful analysis to get higher accuracy of the semantic orientation (Turney & Littman, 2003).

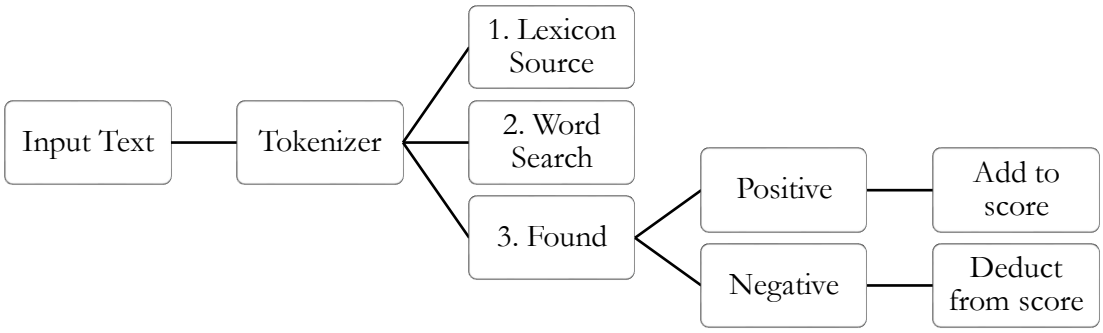

Figure 2. Lexical Sentiment Analysis Approach.

2.4.2 Machine Learning Approach

As introduced earlier, there are three types of machine learning applications depending on the availability of a training dataset 1) Supervised Learning: it is when a sufficient training dataset is available, completely labeled, and done manually by users and domain experts. 2) Semi-supervised: if there is a kind of input is applied to improve the labeled data, or the learning process when dataset labeling is incomplete. 3) Unsupervised: when there is no labeled data, but other methods are used to train the classifier.

Sentiment analysis applications based on machine learning approaches are more likely to be supervised learning by obtaining a labeled dataset to train the learning models. This simplifies the required work and leads to achieving higher accuracy than other methods. When it's difficult to obtain such a dataset, then practitioners attempt to apply semi-supervised learning to help in training the models (Medhat, Hassan, & Korashy, 2014). Also, using models with unsupervised learning to when it works to achieve the goal of the sentiment analysis.

Techniques using supervised learning starts with preparing a training dataset, then doing all necessary data preprocessing on this dataset for the purpose of the sentiment classification task and based on the selected technique. After that, the learning algorithm is trained on this training data (Vohra & Teraiya, 2013). Semi-supervised and unsupervised techniques are used when there is no labeled data available. It is usually used as a support method within a supervised learning approach to help summarize, categorize and tag unlabeled data.

2.4.3 Naïve Bayes (NB)

The NB algorithm is considered a Probabilistic classifier. In sentiment analysis, it computes the net probability of a class based on the distribution of words in the document. It is common to use Bag of Words (BoW) features selection technique to develop the model (Medhat, Hassan, & Korashy, 2014). It has been used in many applications for sentiment analysis for the great success in the classification, and it proved that it performs better than other algorithms especially in multi-label classification (Pang & Lee, 2008).

2.4.4 Maximum Entropy (MaxEnt)

This algorithm uses weights for features that are calculated and combined to select the most likely label for a feature set. The features are encoded into a vector with weights and used to train and test the algorithm (Medhat, Hassan, & Korashy, 2014). MaxEnt algorithm assumes no dependency between features unlike NB algorithm. However, it proves high accuracy and sometimes outperforms NB when the dependency is not affecting the accuracy (Pang, Lee, & Vaithyanathan, 2002).

2.4.5 Support Vector Machine (SVM)

SVM is used widely in text mining applications and sentiment analysis. Its nature allows determining features and the quality of it in a vector of words in a text. SVM uses sematic orientation values of sentences and words in a text to extract and select features for the training the algorithm. It's also used in categorizing and text tagging

(Thakkar & Patel, 2015). This mix of capabilities allows adapting the technique to be used in predicting the sentiment in the text (Medhat, Hassan, & Korashy, 2014).

2.4.6 k-Nearest Neighbors (kNN)

The algorithm tries to calculate the votes of the nearest neighbors and stores them for each instance of the training data. The final decision is done by majority vote of the nearest neighbors of each instance. Then, the class is assigned based on the number of most representatives of neighbors (Aggarwal, Zhai, & editors, 2012).

2.4.7 Decision Trees (DT)

Decision Trees is a supervised machine algorithm, i.e. needs a labeled dataset. The algorithm is usually used in classification and regression machine learning tasks. The main concept of the algorithm is hierarchal division of the data to get rules that leads to the classification. The goal is to predict the class based on the decision rules learned from the divided tree (Aggarwal, Zhai, & editors, 2012).

2.4.8 Random Forest (RF)

It is considered an ensemble of random decision trees (Geurts, Ernst., & Wehenkel, 2006). The class prediction is calculated as the average of the predictions for each decision tree.

More machine learning algorithms can be used in sentiment analysis such as Neural Networks (NN). Algorithms also can be combined together to achieve the goal of sentiment application. For example, (Reyes & Rosso, 2012) used NB, SVM and Decision Trees for aspects extraction tasks that proved high accuracy. The process of supervised machine learning can be illustrated in Figure 3 below.

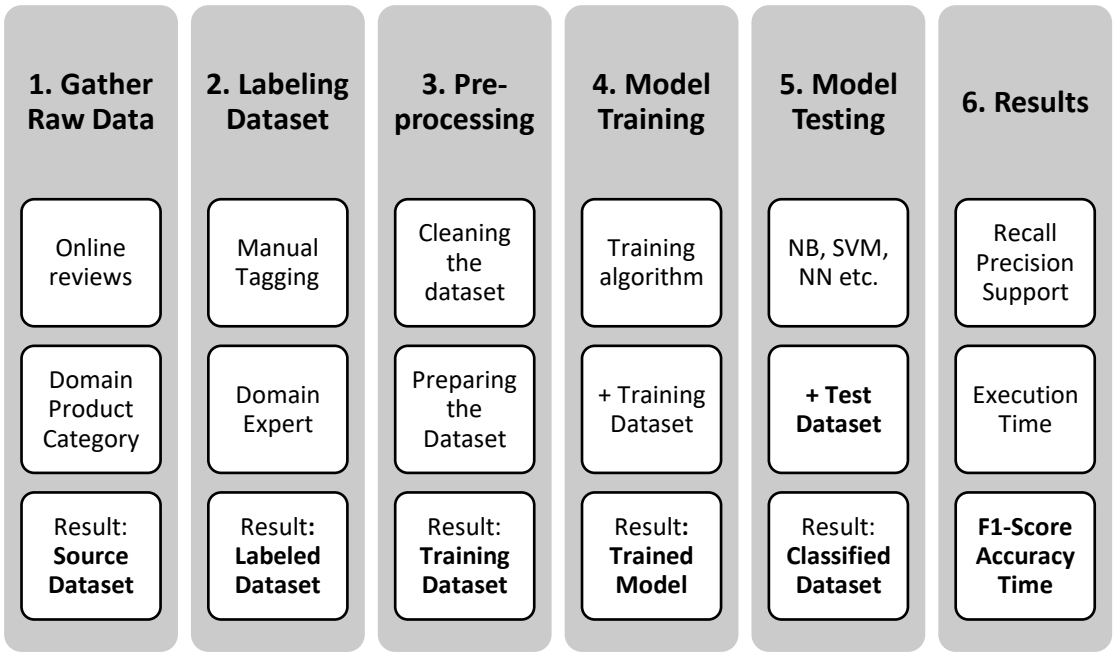

Figure 3. Supervise Machine Learning Approach.

2.5 Binary, Multi-class and Multi-label Classification

In addition to several algorithms, there are different types of applications in machine learning techniques such as Rules Extraction, Clustering, Regression and Classification (Miroslav, 2015). All of it can be used in sentiment analysis, however, classification techniques are the most common to be used. Classification is considered as a predictive task in machine learning. Which means predicting the class of a data record based on the values of other properties in that record in the dataset (Tan, 2006). The prediction is done based on a classification model developed using a pre-classified data. Therefore, classification need supervised learning and a labeled dataset for training the classification model (Alpaydin, 2014).

Classification is mainly done on binary values. It means assigning a class for each data record among two available classes. For example, Win or Lose, Fraud or Authentic, Subjective or Objective. With the continuous development of machine learning, the classification techniques and algorithms evolved to include multi-class and multi-label classifiers. Multi-class indicates that there are more than two classes to choose from. Whereas, multi-label classification – also called multi-label learning – means that one or more of the available classes are assigned to each record (Tsoumakas & Katakis, 2007). Multi-label classification adds a level of complexity on the algorithms and increases the time required for classification (Zhang & Zhou, 2014).

It can be argued that multi-label problem is the same as multi-class problem. However, in multi-class problems the classes are mutually exclusive, which means that each data record is assigned one label only. For example, a review can be either positive, negative or neutral but not two classes at the same time. Whereas, for multi-label problems each label represents a different classification problem. For example, a review could be classified as positive, comparative and aspects-based at the same time. Also, multi-label problems allow having relationships between classes (Liu & Chen, 2015). For example, there could be a correlation between comparative and aspects-based reviews since people tend to compare based on specific aspects, while there will be no correlation between positive and negative reviews in any way.

2.6 Multi-label Classification Techniques

Multi-label classification problems can be addressed in different ways. There are three common multi-label classification techniques, Problem Transformation, Algorithm Adaptation and Ensemble (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012). The following sections give a brief description on these techniques.

2.6.1 Problem Transformation

The most basic technique in addressing multi-label problem is to transform it into one or more binary or multi-class classification problems (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012). This also means transform the problem from multi-label into a single-label one. This can be achieved with three methods. Binary Relevance, Label Powerset and Classifier Chain. These methods are discussed below.

Binary Relevance

The concept behind binary relevance method is to use a One-vs-All (OVA) strategy when dividing the problem into multiple binary classifications. In this method one algorithm is trained per class per label. The average predictions per class label is considered the multi-label prediction (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012).

Label Powerset

This method combines the labels into a set of multi-classes to form a single-label problem. The set of multi-classes generated are distinct. However, the method takes into account possible correlations between labels during implementation. The disadvantage of this method that it could lead to very large number of multi-class combinations when there is a large number of labels. (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012). Therefore, it is noticed during experiment that the multi-classes have a small number of the support metric, i.e. correct positive examples, or what is called number of true positives (TP) in section 2.7.

Classifier Chain

In this method, a chain of binary classification problem is built. The difference between this method and binary relevance that each classification problem includes the predictions from the previous problem, thus the name chain. This way the correlation between labels is taken into consideration. Improvements have also been introduced in classifier chain to reduce the negative examples between the predictions when passed to each chain which leads to better multi-label classification results (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012).

2.6.2 Algorithm Adaptation

It is a technique to extend the machine learning algorithms to be able to handle multi-label classification problems directly without using any problem transformation method. This technique takes advantage of some parameters of the algorithms that can be adjusted or used to classify multiple labels (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012). Some example of adapted algorithms as presented in the literature are: multi-label kNN (MLkNN), multi label Decision Trees (ML-DT) and Rank-SVM (Zhang & Zhou, 2014).

2.6.3 Ensemble

The idea behind ensemble method is taken from both the problem transformation and algorithm adaptation methods. One of the most common ensemble methods is called Random k-Labelsets (RAKEL). RAKEL optimizes and runs each machine learning algorithm on a small random subset of the dataset as a single-label prediction of each element in the power-set of this subset. The method proved good classification results (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012).

2.7 Evaluation Metrics

Most of the evaluation metrics of the classification model are mainly based on the values of the confusion matrix. The confusion matrix is a distribution of the correctly and wrongly predicted examples from the dataset (Tsoumakas & Katakis,

2007). The main values of the confusion matrix are listed below. The term "hits" refers to data records or instances in the dataset.

- True Positive (TP): Number of hits were correctly predicted as positive.
- True Negative (TN): Number of hits were correctly predicted as negative.
- False Positive (FP): Number of hits were wrongly predicted as positive.
- False Negative (FN): Humber of hits were wrongly predicted as negative.

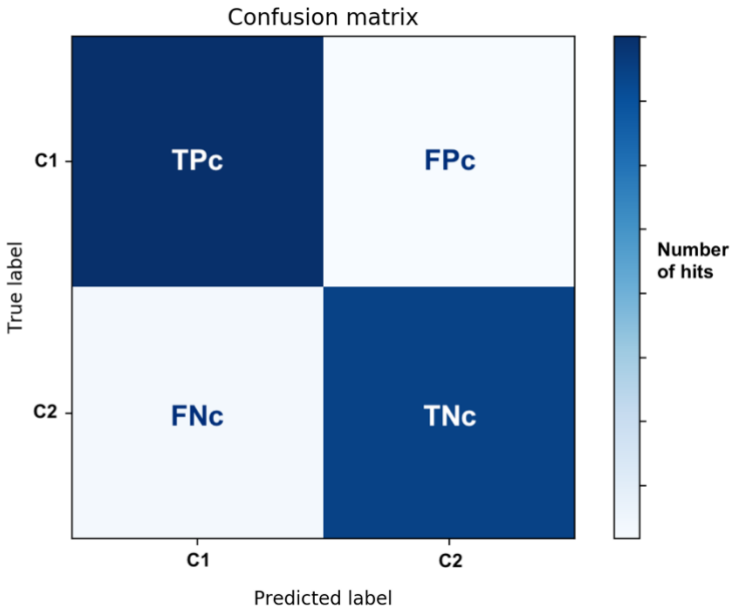

Figure 4. Confusion matrix representation per label (where c = a class in the label).

2.7.1 Precision

Precision is the percentage of correctly predicted hits as positive (TP) over all the predicted hits as positive (TP + FP).

Precision = TP / (TP + FP).

2.7.2 Recall

Precision is the percentage of correctly predicted classes as positive (TP) over all available positive classes, i.e. the correctly predicted hits as positive and wrongly classified as negative (TP + FN).

Recall = TP / (TP + FN).

2.7.3 F1-score

F1-score, sometimes referred to as f-measure or f-score) Is the harmonic mean of precision and recall.

F1-score = 2 x (Precision x Recall) /(Precision + Recall).

F1-score is a good metric when we want to evaluate the balance between precision and recall.

2.7.4 Support

The total the number of hits that were predicted with a specific class.

From Figure 4, we can conclude the following support values:

Support of c1 = FN + TP

Support of c2 = TN + FP

The support measure gives a good indicator if the classes in the labeled dataset is balanced or not. The support values of each class are expected to be close to each other to indicate a balanced distribution of classes.

2.7.5 Accuracy

Accuracy is the percentage of correctly predicted hits as positive and negatives (TP + TN) over all hits.

Accuracy = (TP + TN) / (TP + FP + TN + FN).

In multi-label classification the metrics compute the total accuracy of all labels, i.e. the labels predicted must exactly match the true labels.

2.7.6 Hamming Loss

It is a metric that supports multi-class and multi-label classification problems. It is used to measure the percentage of wrongly predicted labels, by computing the loss or the hamming distance between the predicted and true labels of two set of samples (Tsoumakas, Katakis, & Vlahavas, Mining multi-label data., 2009).

If ***Yj*** is the predicted value for the **label j** of a given sample, ***Xj*** is the corresponding true value, and ***n*** is the number of classes or labels, then the Hamming loss between two set of samples equals to:

$$Hamming\ Loss\ (Xj, Yj) = 1 / n \cdot \sum_{j=0}^{n_{\text{labels}}-1} 1(Xj \mathbin{\vartriangle} Yj)$$

Where Δ means the XOR operation in Boolean logic (Tsoumakas & Katakis, 2007).

Hamming loss equals to 1 – Accuracy for binary classification problems. However, when evaluating a multi-label classification, the case is different. Accuracy is a strict metric that measures correct predictions for all labels. Therefore, errors in each label propagates when measuring the total accuracy, which leads to a lower measurement. Hamming loss metric takes into account partially correct labels when calculating the loss or the hamming distance (Destercke, 2014).

2.7.7 Time Measurement

The labeled dataset used to build the classification model is split into training and testing datasets. Therefore, the time measurement is the total time taken to train the classification model using the training dataset, and to test the classification model using the testing dataset.

3 Research Method and Implementation

3.1 Research Method

The research method for this thesis is a case study of a sentiment analysis application for mining comparative opinions based on product aspects using multi-label machine learning classification techniques. The case study is conducted on a dataset of online customer reviews that is not specific to any domain, product or provider.

The contribution of the case study is evaluating the effectiveness and efficiency of multi-label classification techniques for the sentiment analysis application under study. Due to the diversity of sentiment analysis applications and domains, the evaluation results are not necessarily to be generalized on all similar sentiment analysis applications. This is the justification behind the choice of a case study research method rather than being mainly experimental. (Wohlin, et al., 2012).

The case study consists of two main parts as follows:

1. Design and development of the sentiment analysis application: this part is done by following a design and creation research method, which is a method adopted to describe the design and development of software artifices (Oates, 2006). The method covers the creation of the labeled dataset of online customer reviews and the design of the application in a suitable way for evaluating multi-label classification techniques using Scikit-learn (Pedregosa, et al., 2011). The design and creation process of the application is adapted from the Design Science Research Process (DSRP) (Peffers, et al., 2006). The steps followed in the process are: problem definition, design, development, implementation and evaluation.
2. Empirical experiments: the effectiveness and efficiency of multi-label classification techniques are evaluated based on the performance of different machine learning algorithms when they are used in the technique. Therefore, empirical experiments are conducted on seven machine learning classifiers5. The experimental process adopted in this thesis is suggested by Wohlin, et al. (2012). The steps followed in the experimental process are: scoping, planning, implementation, analysis, and presentation.

The design and development of the application in the first part of the case study is necessary to conduct the empirical experiments in the second part, therefore the application is implemented first. The following sections discuss in detail the method followed in each part.

3.1.1 Sentiment Analysis Application

As mentioned above, the DSRP process suggested by the design and creation research method is used to develop the software artifacts for the sentiment analysis application. The process is illustrated in Figure 5 below.

[5] The term classifier is used instead of algorithm when is practically applied in the experiments, since the term algorithm refers to the algorithm theory rather than actual implementation in machine learning tools.

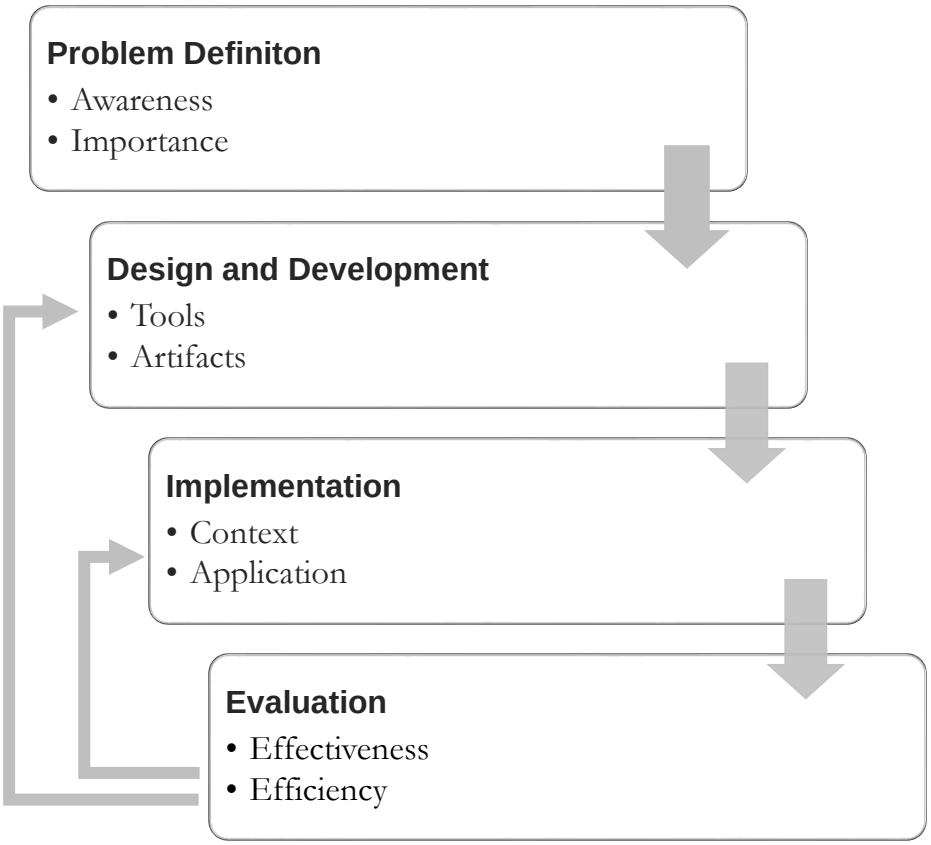

Figure 5. The process to design the application. Adapted from Peffers, et al. (2006).

The process starts with the problem definitions where the elements of the problem are defined in detail. The problem definition includes the awareness and importance of the problem.

The next step of the process is the design and development of the artifacts in the application, including the tools to build the artifacts if needed.

After that, the developed artifacts are implemented in the application and its context in order to achieve the solution to the problem.

Finally, evaluation is performed on the artifact to check if it meets the requirements. If the requirements were not met, necessary changes in the design and development or the implementation are identified.

Problem Definition, Awareness and Importance

The sentiment analysis application under study is the identification of comparative opinions that are based on product aspects in online customer reviews and their sentiment classification. The application includes three classification problems: Identifying comparative opinions, identifying the existence of product aspects, and predicting the sentiment classification of the opinions.

What do these three classification problems mean? What value does they provide in practice? The list of examples below answers these questions.

Example 1:

I bought this laptop two months ago. It's working perfectly and worth buying it. I love it!

The example above indicates a direct opinion by the reviewer on the product. Three phrases indicate a positive opinion in the review "working perfectly", "worth buying" and "love it". In fact, it has been found that verbs, adjectives and adverbs are good indicators for opinions (Benamara, Cesarano, Picariello, Reforgiato, & Subrahmanian, 2007). Therefore, the sentiment classification of the opinion in this review is positive. Objective content is considered irrelevant for the sentiment classification, such as the sentence "bought this laptop before two months".

Example 2:

The zooming in this camera is poor, it only goes up to 10X, but I expected more for the price.

In this review, the opinion is given on the "zooming" as an aspect of the product. It also gives an expectation on the "price" of the product indicating it is high comparing to the zooming aspect. Therefore, the opinion is classified as an aspect-based opinion. Furthermore, the sentiment is negative in two sentences "The zooming in this camera is poor" and "expected more for the price". Which indicates a negative feedback on the product-aspect. This kind of feedback is more helpful for providers and other customers as it gives a justification of the negative opinion.

Example 3:

No issues, this phone charger works better than the original one that comes with the phone in so many ways.

This is a simple comparative review that compares a third-party charger with the original phone charger. The sentence "works better than" is the main identification for comparison. Words such as "better" and "than" indicates a comparative opinion. Therefore, the review can be classified as comparative. Also, it can be noticed that the sentiment classification of the opinion is positive, which means that the product is better compared to the other product mentioned in the review. This information is very useful for businesses to understand their competitors and gives a better idea for other customers when reading the review.

Example 4:

I like the new look and feel of the phone. Really nice upgrade, it has a better resolution and battery life than the previous model.

The review above is given by comparing one product aspects with same aspects of another product. The aspects compared are the look and feel, resolution and battery life in the new model versus the old model. This means the opinion in the review is classified as both comparative and aspects-based. Also, the sentiment of the opinion is positive, thus the product aspects compared are better. This makes the feedback in the review more valuable for the companies and helps other customer in making a better decision when buying the product.

Awareness:

It can be concluded from the examples above that the problem has three binary classifications problems, i.e. each problem has two classes as positive or negative, comparative or non-comparative, and aspects-based or not. However, three classes are assigned in total to each review. Each combination of binary classes is considered as one label. Therefore, the problem is considered a multi-label classification problem from machine learning perspective. The labels are: Comparative review (Binary: Yes, No), Aspects-based review (Binary: Yes, No), and the Sentiment Classification of the review (Binary: Positive, Negative).

The problem can be addressed by using different multi-label classification techniques such as problem transformation into multiple binary and multi-class classification problems. This requires developing a multi-label classification model that uses a machine learning classifier. The classifier needs to be trained and tested using a labeled dataset of online customer reviews. The dataset has to be prepared and preprocessed in order to achieve better classification performance.

Importance:

It is noticeable from the examples that the defined problem has a high level of importance from both business and research point of views. From a business point of view, the importance can be identified as:

1. Providing useful insights to companies about their products, especially when compared with other products.
2. It helps businesses to maintain their competitive edge by addressing the aspects of their products that are reviewed and compared.
3. For e-commerce platforms, it provides valuable information for their new customers by showing useful reviews to them.

From a research point of view, the importance can be identified as:

1. Finding a suitable solution for the sentiment analysis application under study.
2. Evaluating new techniques in the research field of sentiment analysis.
3. Providing an idea about the effectiveness and efficiency of multi-label machine learning techniques when they are used in this type of sentiment analysis applications.

As a summary, the definitions of the main elements in sentiment analysis application are listed in Table 1 below with prefix (E#) for each.

Prefix	Element	Definition
E1	The Application	Identifying comparative opinions that are based on products aspects and their sentiment classification in online customer reviews.
E2	Sentiment Classification	The classification of the given opinion as Positive or Negative (Turney, 2002).

E3	Comparative Opinion	An opinion that expresses similarities or differences between two or more objects (Jindal & Liu, 2006).
E4	Product Aspects-based	The occurrences of any product aspects, features, functions, properties, entities etc., in the review (Pang & Lee, 2008)
E5	Classes and Labels	Sentiment polarity, comparative opinion and product aspects-based are the labels that need to be assigned to each review. Each label consists of two binary classes (Positive/Negative, Yes/No).
E6	Classification Problem	Assigning the correct classes that are targeted in the application for each review.
E7	Binary Classification	Assigning one class among two available classes for each review, i.e. single-label is assigned.
E8	Multi-class Classification	Assigning one class among multiple available classes for each review, i.e. a single-label is assigned.
E9	Multi-label Classification	Assigning multiple classes among multiple available classes for each review, i.e. multiple labels are assigned.
E10	Machine Learning Classification Technique	The classification technique that is used to implement the multi-label machine learning approach used in the application.
E11	Machine Learning Classifier	The implementation of the machine learning algorithm to be used in the classification technique.
E12	Classification Models	The implementation of the machine learning techniques and classifiers used in the application.
E13	Labeled Dataset	A dataset that contains labeled online customer reviews to be used in building the classification model. The model will learn from the labeled data so that it can classify new unlabeled data.
E14	Data Preparation and Preprocessing	The steps that need to be performed to clean the review text in the labeled dataset and transform it into a suitable format for the classification model.
E15	Classification Performance	The accuracy of the classification model in assigning labels and classes to the reviews. Also, the time needed for training and testing the classifier.

Table 1. Elements of the sentiment analysis application and their definitions

Application Design and Development

The elements that need to be designed and developed for the application are listed below as per there codes in Table 1:

1. Prepare a labeled dataset (E13): including the processes for selecting the source dataset and filtering it into a development dataset that is more suitable for manual labeling.
2. Data preparation and preprocessing (E14): cleaning the labeled dataset and preparing it for the classification model.
3. Design of the labels and classes (E5): designing and transforming the labels and their classes in a suitable way for multi-label machine learning.
4. Developing the classification model (E12): including features selection technique and preparing the classifier to be used with multi-label classification techniques.

The artifacts of the sentiment analysis application are the labeled dataset and the software that is used to implement the application. The software includes the functions for preprocessing the labeled dataset and building the classification model. The software builds the classification model in a suitable way so that it can be experimented using different multi-label classification techniques and machine learning classifiers.

The design of the application elements and artifacts is discussed in the following sections. Then, the implementation and evaluation of the artifacts are discussed in the implementation section (3.2).

3.1.2 Labeled Dataset

The process of creating a labeled dataset is divided into three main stages. First, selecting the source dataset to get online customer reviews. Then, preparing a development dataset that contains filtered reviews that are suitable and ready for labeling. Finally, labeling the reviews in the development dataset by humans to create the labeled dataset.

Selecting a source dataset

The context of the sentiment analysis application is online customer reviews. A suitable source dataset for this context is a dataset of online customer reviews for the products on Amazon website. The dataset is made available on this public link for researchers http://jmcauley.ucsd.edu/data/amazon/ (He & McAuley, 2016).

The dataset was originally used in the paper "Addressing Complex and Subjective Product-Related Queries with Customer Reviews" (He & McAuley, 2016) In their paper, the authors discuss a binary classification model to predict if a review can answer a question asked by another Amazon customer.

The reviews in the source dataset were filtered from any fake, abusive and spam reviews. Also, all duplicates were removed. Each record in the source dataset contains meta raw data for the review. The main attributes of the review in the source dataset are as follows:

1. Review summary: up to 128 characters' length written by the reviewer as a summary.
2. Review text: up to 30000 characters' length, and this represents the full review.
3. Overall (rating): the product rate of a range between 0-5 (starts) given by reviewers themselves.
4. Helpful: the number of helpful votes by real users.

The dataset contains separate packages for each product category on Amazon. Each package contains a compressed JSON file for the reviews in the category. Many categories are given in the dataset. Therefore, a selection of some categories was necessary. The design of the selection process is described below.

Design properties:

1. Domain independent: Selected among 5 categories of products: Electronics, Clothing & Shoes, Apps for Mobile, Cellphones and Accessories, and Health & Personal Care.
2. More likely to contain comparative opinions: the categories were selected on the basis that they are more likely to have comparative and aspect-based reviews while making sure it is still domain independent.

The size of the selected categories of the source dataset is 3,600,000 reviews.

Preparing a development dataset

It is a subset of the source dataset that is more suitable for labeling.

Design properties:

1. Size: 200000 reviews.
2. Review length: is between 200-800. The minimum is set to 200 to get meaningful reviews since short reviews have less information. The maximum is set to 800 characters to reduce the required efforts when manually labeling the reviews by humans.
3. Review attributes: only the Review Summary, Review Text and Rating attributes are taken into consideration. Other attributes such as the reviewer ID, name and time are skipped.
4. Distribution: the estimated distribution of the reviews in terms of the rating is decided to be 120,000 of positive reviews, and 80,000 of negative reviews. The distribution is designed to make sure we get equal distribution for comparative reviews which tend to be more in the negative reviews.

Implementation steps:

1. The source dataset is in JSON format and needs parsing into a more suitable format for the data preprocessing such as CSV.
2. The JSON source files were read and converted to CSV using Python.

3. A special code is written to select reviews randomly for two ratings ranges:
    a. Positive reviews: 4-5 stars rating (80000 reviews)
    b. Negative reviews: 2-1 stars rating (120000 reviews)
4. The review text length has been divided into three ranges as well; 200-400 characters, 400-600 characters and 600-800 characters. Then, an equal number of reviews are selected for each range, about 33% each, to get an equal distribution.
5. The development dataset is loaded into a MySQL database so that it can be used for labeling the dataset in next step.

Labeling the dataset

It is a critical step in this case study since the training dataset is labeled manually. The reviews were labeled with the assistance of three students at Jönköping University in Sweden. It is assumed in this thesis that it is intuitive for humans to classify a review whether it is positive, negative, containing comparative opinion and if it has product-aspects. However, it is more correct to claim that the labeling is done from a customer or user perspective rather than the perspective of a domain expert or a product provider, which could be a future work investigation.

Design properties:

1. Size: 20,000 reviews, which is equal to 10% of the development dataset.
2. Labeling: manual labeling from customers or user perspective. The pre-given rating label by the reviewer in the source dataset is taken into consideration for the sentiment class. If the rating is equal to 4 or 5, the sentiment class of the review is pre-labeled as Positive. If the review rating is equal to 1 or 2, the sentiment class is considered Negative. For the comparative and aspects-based labels, the labeling is done by identifying words, phrases or sentences that indicate these labels.
3. Subjectivity consideration: relevant sentences that were identified in the previous step are highlighted and saved in the database. These sentences help in evaluating and data preprocessing the dataset.
4. The dataset is balanced:
    a. Equal distribution between negative and positive classes with 50% of total reviews for each.
    b. Balanced distribution between non-comparative and comparative classes with no class is less than 40% and more than 60% of reviews. This because it was difficult to get 50% distribution between both during manual labeling.
    c. Fair distribution between non-aspects and aspects where no class is less than 35% and more than 65% of reviews. This percentage is decided because the number of reviews that are aspect-based is larger than the number of non-aspect-based reviews.

Implementation steps:

1. A simple web page was developed for labeling the dataset. It reads the reviews from the development dataset that was loaded into the database. Then, shows the review text, labeling options and fields for filling the identified sentences that indicates each label.
2. The web page is selecting a review randomly from the development dataset.
3. The web page can be configured to show positive or negative reviews only. This is needed to get an equal distribution for these classes in the dataset.
4. The link to the web page was given to the students for labeling.
5. Reviews that cannot be decided during labeling are just skipped.
6. The labeled reviews during implementation are stored in a separate table in the same MySQL database where the development dataset is loaded.
7. After the labeling is done, the dataset is exported into a CSV file which is more suitable for the classification in the application software.

Evaluation steps:

1. Labeling was divided into milestones of 1000 reviews each.
2. After each labeling milestone, one student evaluates the labeled reviews.
3. The role of the evaluator changes among the students in each milestone.
4. In case if the evaluator disagrees with the labeling, a discussion is made between the three students to agree on correct label. Some reviews are also decided to be skipped during evaluation.
5. Obvious mistakes in labeling are corrected by the evaluator without discussion.
6. The size of the labeled dataset is evaluated after each milestone using binary classification on each individual label. The evaluation is done by checking the recall and support of classes so that the classification problem is balanced. Also, the accuracy was monitored to decide the feasible size of the labeled dataset. This means if the classification accuracy is low in the single classification problem, then the size needs to be increased. The dataset size of 20,000 was decided after several evaluation iterations.
7. Statistics page was developed to show the classes distribution in the dataset.
8. Assuring that the dataset is balanced by adjusting the labeling process. This is done by monitoring the distribution in the statistics page and configuring the web page to show only positive or negative reviews when needed.

The application software includes all the functions required for labeling the dataset. It also includes the code for exporting the labeled dataset to CSV when it is ready in the database. These functions are not the focus of the research. Therefore, they are not discussed in detail in this thesis.

3.1.3 Data Preprocessing

The source dataset of online customer reviews from Amazon is cleaned from duplicates, abusive and irrelevant reviews. Also, the emojis, codes and links are removed. This reduces the required data preprocessing on the labeled dataset. The remaining data preprocessing can be divided into two parts, data cleaning and data preparation.

Data Cleaning

The data cleaning tasks are:

1. Removing of unnecessary special characters: only these special characters are kept in the review text ". , ' - _ $ % ? !" because they may have a meaning in the sentence. All other special characters were removed.
2. Remove extra punctuation and white spaces: extra punctuation such as replacing "…" with ".". This is needed as it may affect the tokenization in data preparation.
3. Process accented characters: replace letters with accent notations such as "ê ä ó" with the normal English letters "e a o". These characters may lead to different variations of a word. This can be an influencing factor when building the classification model.
4. Handling contractions: replacing words such as "I'd" with "I would / I had". A tool called "pycontractions" was used for this purpose. In fact, the tool uses machine learning to select the correct replacement (PyPI, 2018).

Data Preparation

The data preparation tasks are:

1. Tokenization: it means splitting the review text into tokens of sentences, then splitting each sentence into tokens of words. The tool is implemented in the Natural Language Toolkit (NLTK) (Bird, Loper, & Klein, 2009).
2. Subjectivity classification**:** identifying subjective sentences that have an opinion in them. Then, these sentences are used in the training and testing datasets and other sentences are removed. The tool used for this purpose is called VADER introduced by (Gilbert, 2014).
3. Subjectivity of comparative and aspect-based sentences: the sentences that were identified as indicators for comparison and aspects-based are assumed subjective. Therefore, they are not removed by the subjectivity classification task in step 3 if there is no sentiment weight for it.

Practitioners in sentiment analysis and text classification apply other steps for data preprocessing such as negation handling, which is adding the tag NOT_ to the word when it is negated. Also, stemming which means replacing the word with its stem root in English. However, both techniques have negative effects on the classification performance when evaluated by using binary classification models. Therefore, they were not included in data preparation for this sentiment analysis application.

3.1.4 Design of Labels and Classes

The labels and classes need to be designed and transformed into a suitable format for the classification problem. As introduced earlier, the labels are: Comparative, Aspects-based and the Sentiment. Each single label has two classes as binary values, that usually have the values (0/1), (Yes/No), (True, False) or (Positive/Negative). In this application the values (0/1) are used for simplicity. Therefore, there are six different classes. These labels and classes are listed in Table 2 below with prefixes (L#) for labels and (C#) for binary classes.

Prefix	Label Name	Binary Class Prefix - Name	Value
L1	Sentiment Classification	C1 - Negative	0
		C2 - Positive	1
L2	Comparison Identification	C3 - Non-comparative	0
		C4 - Comparative	1
L3	Aspects-based Identification	C5 - Non-aspects-based	0
		C6 - Aspects-based	1

Table 2. Classification labels and the binary classes

It is necessary to convert the values of the labels and their classes into a suitable format for multi-label classification. The format is different than the value of the binary classes above. One of the methods in Scikit-learn, called MultiLabelBinarizer, transforms a set of classes from multiple labels into a binary matrix. The matrix indicates the presence of a class by (1) and the absence by (0) for a specific label. This is one of the recommended methods for passing the labels to the classifier in Scikit-learn (Scikit-learn, MultiLabelBinarizer, 2018).

For example, it can be noticed in Table 2 that there are 3 labels with 6 possible classes (C1, C2, C3, C4, C5, C6). For each label, only one class is assigned. Assuming that for a specific data instance, the classes assigned from each of the available labels are: C1, C4, C6. Then, by following the multi-label binarizer approach, the presence of the class is marked with 1 whereas the absence is marked with 0. Based on that, the value of the label will be (1, 0, 0, 1, 0, 1).

The multi-label representation for the classes is shown in Table 3 below with prefix (ML#) for each.

Prefix	Multi-label Representation	Class Code	Value
ML1	Positive, Comparative, Aspects-based	C1, C3, C5	101010
ML2	Positive, Comparative, Non-aspects-based	C1, C3, C6	101001
ML3	Positive, Non-comparative, Aspects-based	C1, C4, C5	100110

ML4	Positive, Non-comparative, Non-aspects-based	C1, C4, C6	100101
ML5	Negative, Comparative, Aspects-based	C2, C3, C5	011010
ML6	Negative, Comparative, Non-aspects-based	C2, C3, C6	011001
ML7	Negative, Non-comparative, Aspects-based	C2, C4, C5	010110
ML8	Negative, Non-comparative, Non-aspects-based	C2, C4, C6	010101

Table 3. Multi-label class representation and their binary values

3.1.5 Classification Model

Development of the classification model includes five main tasks as follows:

1. Reviews Text and Labels: the labeled dataset has four columns. One for the reviews text and three for the labels. In practice, when building the classification model, the reviews text column is stored in one array, while the labels columns are stored in a separate one. The separation is needed for the selection of features in step 3.
2. Sampling and splitting: the labeled dataset need to be split into two datasets for training and testing the classification model. Usually, the training dataset size is 75% and the testing dataset size is 25% of the total labeled dataset. Different sampling methods for selecting the reviews for each dataset can be used, such as random and stratified sampling. The sampling is done on both arrays of reviews text and labels.
3. Features selection: the reviews text array in the training dataset will be indexed as a vector of words using BoW technique. This is usually done by identifying words with higher frequencies in the whole set of the reviews and selecting most frequent words. These words are called features. The array, which consists of one column, will be transformed into a number of columns. The number equals the total number of features.
4. Training and developing the classification model: the array of selected features and the labels array of the training dataset are given as an input to the classification model. The machine learning algorithm learns the labels from the features and generates a model that can predict the labels for new unlabeled reviews.
5. Using multi-label classification technique: the multi-label classification technique is used as a wrapper around the classification model using the techniques available in (Scikit-multilearn, 2018).
6. Evaluation of the classification model: the evaluation of the classification model is done by predicting the classes of the reviews text array of the testing dataset. Then, scoring tools are used to compare the predicted labels with the actual labels array for the testing dataset. These tools report a set of metrics that are used for the evaluation such as accuracy, precision, recall and F1-score.

3.1.6 Application Software

Python programming language is used to develop the application software. Latest version of python 3.7 is decided to be used. The machine learning tools used is Scikit (Machine Learning in Python, 2018). The tool used for data preprocessing is NLTK 3.7 (NLTK, 2018). The functions of the software are:

1. Load the source dataset: the code to read the CSV file of the source dataset and load it to the database. The code to convert the source files from JSON to CSV is given in the dataset link, so it was not included in the software.
2. Prepare development dataset: the code that prepares the development dataset from the loaded source dataset and stores it in a database for manual labeling.
3. Export labeled dataset: the labeled dataset is exported from database and transformed to a CSV format after labeling is completed.
4. Preprocess labeled dataset: cleaning and preparing the labeled dataset for the classification model.
5. Create dataset arrays: the preprocessed labeled dataset is stored into two arrays of reviews text and labels.
6. Split and sampling function: to split the labeled dataset using a sampling method into training and testing.
7. Transform labels: transform the labels and their classes into a suitable form for multi-label machine learning classifier.
8. Features selection: a function to extract the features from the training dataset based on BoW technique.
9. Classification model: including the implementation of classifier and the multi-label classification technique so that they are ready for training, prediction and testing.
10. Evaluation: the code for testing and evaluating the classification model.

The web page developed for labeling the reviews is designed using HTML and PHP. The code for it is given in the application software repository given in the implementation section 3.2.1). However, it is not discussed in this thesis.

3.1.7 Multi-label Classification Techniques

There are three main multi-label classification techniques described in section 2.6, problem transformation, algorithm adaptation and ensemble. It has been found that the evaluation of those techniques depends the availability of reliable practical implementations and tools for them. (Szymański & Kajdanowicz, 2017) have a fair implementation of these methods and techniques in Scikit-multilearn tool (Scikit-multilearn, 2018). These implementations are sufficient to achieve the purpose of this case study, which is an evaluation of the techniques rather than comparison or finding an optimum solution.

The problem transformation techniques have been fully implemented and used by practitioners. For the algorithm adaptation technique, it is only possible to evaluate the algorithms that have practical implementation of the adaptation such as

MLkNN. For ensemble, RAKEL method has been implemented and used with different algorithms. Therefore, it is used for evaluating this technique. Therefore, the term (RAKEL) is added to the name of the technique.

The multi-label classification techniques evaluated in the experiment are listed in Table 4 with prefix (CT#). Furthermore, the multi-label classification techniques are considered the first control group in the experiments.

Prefix	Technique Name	Description
CT1	Problem Transformation	Transforming the multi-label classification problem into multiple binary or multi-class classification problems.
CT2	Algorithm Adaptation	Adapting machine learning algorithms to handle multi-label classification problems.
CT3	Ensemble (RAKEL)	Combining both techniques above an ensemble of optimized machine learning algorithms to handle multi-label classification problems.

Table 4. Multi-label classification techniques (Experimental control group 1)

The problem transformation technique, described in section 2.6.1, is done using three different methods listed in Table 5 below in with the prefix (PT#). These methods are considered the second control group in the experiments.

Prefix	Problem Transformation Method Name	Description
PT1	Binary Relevance	Ensemble of classifiers. Assumes no dependency between labels.
PT2	Label Powerset	Considers every unique combination of labels as a single label or multi-class. Dependency between labels is counted.
PT3	Classifier Chain	A chain of binary classifiers. Dependency between labels is counted.

Table 5. Problem transformation methods (Experimental control group 2)

3.1.8 Machine Learning Classifiers

The experimental machine learning classifiers, described in section 2.4, are listed in Table 6 below with their prefix as (MC#) and short codes. The machine learning classifiers are considered the third control group in the experiments.

Prefix	Codes	Machine Learning Classifier Name
CL1	NB	Naïve Bayes Classifier
CL2	SVM	Support Vector Machine Classifier
CL3	MaxEnt	Maximum Entropy Classifier, or Logistic Regression as called in some machine learning tools such as Scikit-learn (LogisticRegression, 2018) (Mount, 2011).
CL4	kNN	k-Nearest Neighbors Classifier
CL5	DT	Decision Tree Classifier
CL6	RF	Random Forest Classifier
CL7	MLkNN	Multi-label k-Nearest Neighbors Classifier

Table 6. Machine learning classifiers (Experimental control group 3)

Scikit-learn applied a problem transformation implementation to kNN, DT and RF classifiers to support multi-label classification out of the box (Scikit-learn, Multiclass and multilabel algorithms, 2018). It uses techniques such as binary relevance and multi-class problem transformation using One-vs-All mechanism. This is not listed as a separate method under problem transformation techniques since the method is not exactly specified in the documentations of the tool. Therefore, those classifiers are experimented directly under the problem transformation technique directly using this implementation.

Classifier Optimization

Each machine learning classifier has a set of parameters to be configured when it is used in the classification model. For example, Maximum Entropy classifier has a parameter called Tolerance (tol) for specifying the stopping criteria (Scikit-learn, LogisticRegression, 2018). Changing this parameter may change the classification results. Optimizing the parameters to be used in each classification model is not in the scope of this case study.

Therefore, the default parameters values are used when applicable, unless it is necessary to adjust the default values when significant change in the performance is expected. In this case of adjusting parameter default values, separate trials are done during the empirical experiments to understand the change effect. These trials are not documented. However, the parameters are listed in the implementation section, so that they are documented in the thesis.

Furthermore, some classifiers work with certain transformation methods for the input data that are different from the methods used for other classifiers. The parameters and transformation configurations for each algorithm in the experiments are listed in top of each experiment in the implementation section.

3.1.9 Empirical Experiments

The experimental process followed in this thesis is taken from the book of *Experimentation of Software Engineering* by Wohlin, et al. (2012). The steps followed in the process are: scoping, planning, implementation, analysis, and presentation.

Experiments Scope

The experiment scope defines the goal of it (Wohlin, et al., 2012). Therefore, the template for the experiment scope is done using the goal template by (Basili & Rombach., 1988).

> Analyze <Object(s) of study>,
> for the purpose of <Purpose>,
> with respect to the <Quality focus>,
> from the point of view of the <Perspective>,
> in the context of <Context>

Defining the elements of the scope as in the template above is done by answering the following questions:

Q1: What is studied?
A: 1) The sentiment analysis application as defined in section 3.1.1.
2) The multi-label classification techniques in Table 4.

Q2: What is the intention?
A: Evaluating these techniques when used in the sentiment analysis application.

Q3: Which effect is studied?
A: The effectiveness and efficiency of each multi-label machine learning technique, based on the performance of the classifiers in Table 6, when used in the technique.

Q4: From what perspective?
A: From a research point of view.

Q5: What is the context of the study?
A: In the context of a master's thesis work.

Summarizing the answers above the scope template of the experiments is:

> Analyze the defined sentiment analysis application and the multi-label machine learning classification techniques,
> for the purpose of evaluation,
> with respect to their effectiveness and efficiency,
> from the point of view of the researcher,
> in the context of a master's thesis work.

Experiments Planning

The most important parts of the experiments that need to be planned for this thesis work are the variables and the tools used to conduct the experiments.

For the variables, there are the *independent variables* that can be controlled and changed in the experiment, and the *dependent variables* that represent the measurement of the effect under study.

The independent variables for the experiments in this research are annotated by the prefix (IV#). For variables that have multiple experimental options, a letter is added next to the number as #a, #b and so on. The options prefix is also added form the experimental group itself. The variables are defined in Table 7 below:

Prefix	Name	Description
IV1	Dataset Size	The total number of records in the labeled dataset.
IV2	Training Dataset Size	The size of dataset partition that is used to training the classification model.
IV3	Test Dataset Size	The size of dataset partition that is used to test the classification model.
IV4	Sampling Method	The sampling method used when splitting the dataset into training and testing dataset.
IV5	Cross Validation Folds	The number of iterations of running the classification model to get the average scores.
IV6	Features Selection Method	The features selection method used to create the dataset features for building the classification model.
IV7	Number of Features	The total number of features extracted using the features selection method above.
IV8	Minimum Number of n-grams	The minimum number of words in the features selected.
IV9	Maximum Number of n-grams	The maximum number of words in the features selected.
IV10	Classification Technique	Three options - Control group 1:
		CT1 - Problem Transformation
		CT2 - Algorithm Adaptation
		CT3 - Ensemble (RAKEL)
IV11	Multi-label Problem Transformation	Three options - Control group 2:
		PT1 - Binary Relevance

		PT2- Label Powerset
		PT3 - Classifier Chain
IV12	Machine Learning Classifier	Seven options - Control group 3:
		NB - Naive Bayes
		SVM - Support Vector Machine
		MaxEnt - Maximum Entropy
		kNN - k-Nearest Neighbors
		DT - Decision Tree
		RF - Random Forest
		MLkNN - Multi-label k-Nearest Neighbors

Table 7. Independent variables for the experiments

It is common in empirical experiments to have one dependent variable to be measured after setting all the independent variables. In this research, the main dependent variable is the performance. However, there are several sub-variables that are needed to give the minimum understanding of the algorithm performance (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012).

Therefore, the performance dependent variables using (DV#) prefix are listed in Table 8 below.

Prefix	Name	Variable Rationale
DV1	Support	Indicates if the classification distribution is balanced among the classes in the dataset.
DV2	Precision	To evaluate the specific prediction of each label.
DV3	Recall	To evaluate the predictions coverage for each label.
DV4	F1-score	Used in the analysis to combine both precision and recall which are given for completeness.
DV5	Macro Average of Precision	The average of precisions for all the labels, measured for completeness.
DV6	Macro Average of Recall	The average of recall for all the labels, measured for completeness.
DV7	Macro Average of F1-score	A common metric for evaluating the multi-label performance when the dataset is balanced (Madjarov, Kocev, Gjorgjevikj, & Džeroski, 2012).
DV8	Hamming Loss	Percentage of labels that are incorrectly classified.

DV9	Accuracy	Measure the accuracy for binary classification for evaluation puposes of the dataset. Used in the performance analysis for completeness (Found that it has a strong correlation with other variables).
DV10	Total Time	Total time for training and testing the classifier.

Table 8. Dependent variables for the experiments

TP = True Positives, TN = True Negatives, FP = False Positives, FN = False Negatives, ALL = All available records. P = Precision, R = Recall, c = class number, n = total number of classes.

The instruments and tools used in the experiment are listed in Table 9 below with prefix (TI#).

Prefix	Name	Value
TI1	Programming Language	Python 3.7
TI2	Machine Learning	Scikit-learn v0.20.1 Scikit-multilearn v 0.1.0
TI3	Test Machine	MacBook Pro (Retina, 15-inch, Mid 2014)
		Processer: 2,5 GHz Intel Core i7
		Memory: 16 GB 1600 MHz DDR3
		OS: macOS Mojave 10.14
		Disk: 500 GB Flash Storage (SSD)
TI4	Test Environment	Terminal in macOS 10.14
TI5	Test Directory	The local path to the project folder, on the test machine in TI3.

Table 9. Experiments tools and instruments

Experiments Implementation

There are three experimental control groups that drive the implementation of the experiments. The techniques in Table 4 represent experimental control group 1. The three methods for the first technique CT1 - Problem Transformation as in Table 5 represent experimental control group 2. The techniques and methods are applied to the classifiers in Table 6 which represents experimental control group 3.

The problem transformation methods are applied to the first 3 classifiers NB, SVM and MaxEnt. For kNN, DT and RF classifiers, they are experimented only once under the Problem transformation technique directly. MLkNN is also experimented once under CT2 - Algorithm Adaptation technique. CT3 - Ensemble (RAKEL) technique is applied to first three classifier as well. Therefore, the implementation can be divided into four stages as follows:

Stage 1:

In this stage, the three methods, PT1, PT2, PT3 in problem transformation technique, CT1, is applied the first 3 algorithms in experimental group 3. This leads to 9 experiments to be performed in this stage.

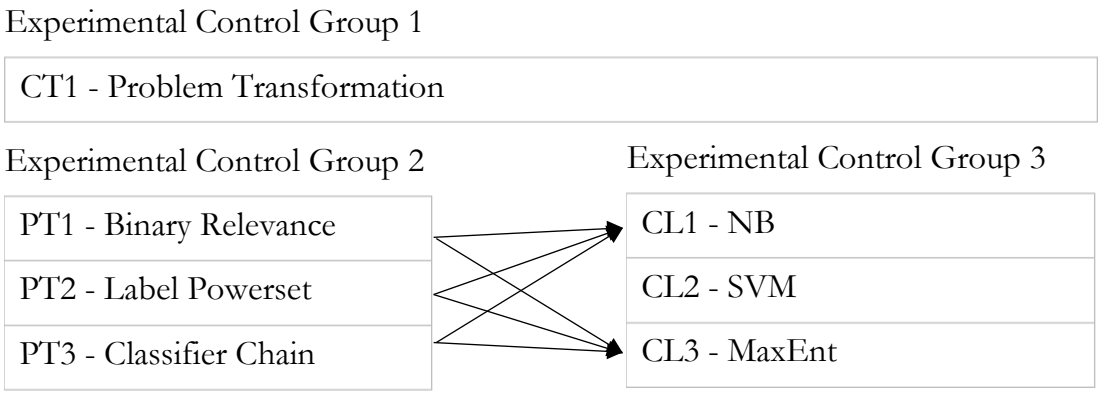

Figure 6. Stage 1 of experiments.

Stage 2:

In this stage, classification technique, CT1, is applied to the kNN, DT and RF classifiers only in experimental control group 3. This leads to 3 experiments to be done for this stage.

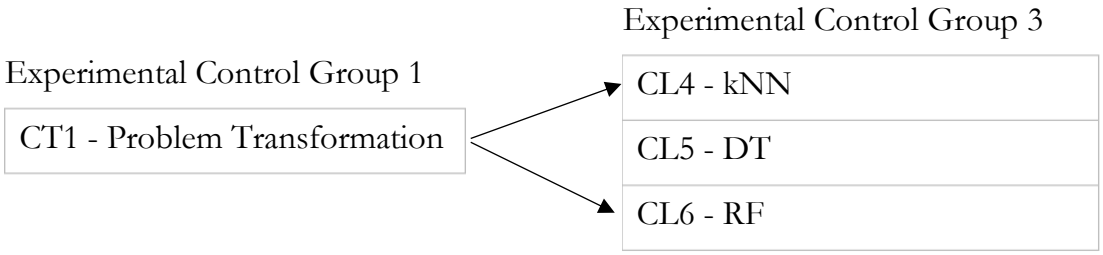

Figure 7. Stage 2 of experiments.

Stage 3:

In this stage, the algorithm adaptation technique, CT2, is applied only to MLkNN, from experimental group 3, so one experiment is performed for this stage.

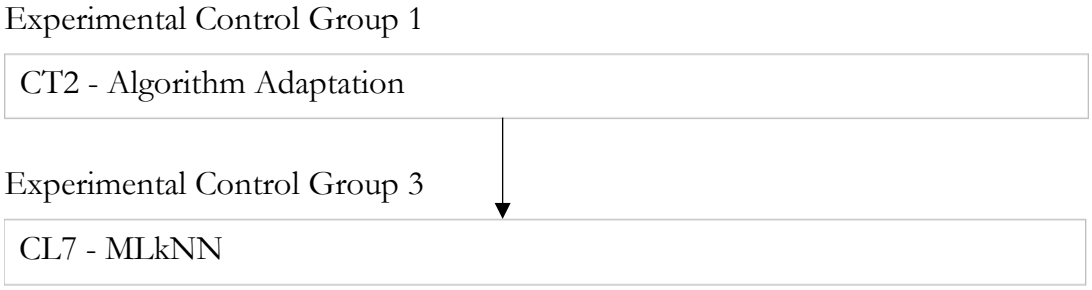

Figure 8. Stage 3 of experiments.

Stage 4:

For the option CT3 - Ensemble, it is applied using RAKEL method on kNN, DT and RF. Therefore, there are 3 experiments to be implemented in this stage.

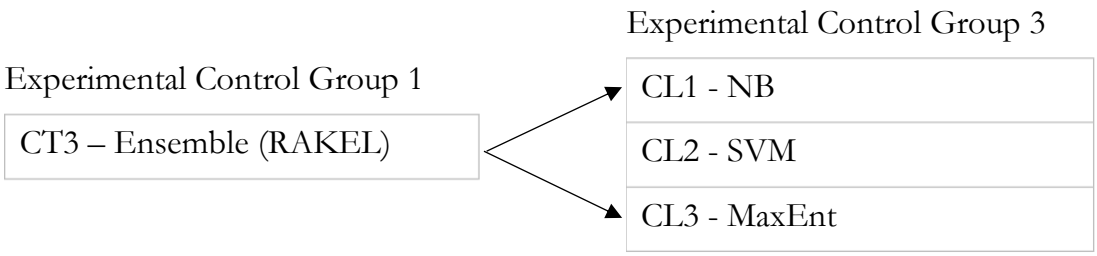

Figure 9. Stage 4 of experiments.

In total, there are 16 experiments to be implemented among the four stages of experiments. Each experiment with its prefix (EX#) and their independent variables configurations is listed in Table 10. The independent variables from 1 to 9 are assumed fixed and not included in the configurations field.

Prefix	Name	Configurations
EX1	Experiment 1	IV10 = CT1, IV11 = PT1, IV12 = CL1 - NB
EX2	Experiment 2	IV10 = CT1, IV11 = PT1, IV12 = CL2 - SVM
EX3	Experiment 3	IV10 = CT1, IV11 = PT1, IV12 = CL3 - MaxEnt
EX4	Experiment 4	IV10 = CT1, IV11 = PT2, IV12 = CL1 - NB
EX5	Experiment 5	IV10 = CT1, IV11 = PT2, IV12 = CL2 - SVM
EX6	Experiment 6	IV10 = CT1, IV11 = PT2, IV12 = CL3 - MaxEnt
EX7	Experiment 7	IV10 = CT1, IV11 = PT3, IV12 = CL1 - NB
EX8	Experiment 8	IV10 = CT1, IV11 = PT3, IV12 = CL2 - SVM
EX9	Experiment 9	IV10 = CT1, IV11 = PT3, IV12 = CL3 - MaxEnt
EX10	Experiment 10	IV10 = CT1, IV11 = N/A, IV12 = CL4 - kNN
EX11	Experiment 11	IV10 = CT1, IV11 = N/A, IV12 = CL5 - DT
EX12	Experiment 12	IV10 = CT1, IV11 = N/A, IV12 = CL6 - RF
EX13	Experiment 13	IV10 = CT2, IV11 = N/A, IV12 = CL7 - MLkNN
EX14	Experiment 14	IV10 = CT3, IV11 = N/A, IV12 = CL1 - NB
EX15	Experiment 15	IV10 = CT3, IV11 = N/A, IV12 = CL2 - SVM
EX16	Experiment 16	IV10 = CT3, IV11 = N/A, IV12 = CL3 - MaxEnt

Table 10. Experiments and their variables configurations

The tools and instruments in Table 9, are utilized to run the experiments. In order to run any of experiments in Table 10, the module "run.py" is used. The module file is included in the repository link of the application software, which is given in the implementation section (3.2.1). The commands below are used in the execution terminal, where the <directory> parameter is the path and the <experiment> parameter is the experiment prefix.

```
<directory>: python run.py <experiment>
```

Examples:

```
Desktop: python run.py EXP3
```

Experiments Presentation

The results of each experiment are presented in terms of the values of their dependent variables after implementing the experiment. The template in the table below is used to present the experiment results.

Label	Class	Precision	Recall	F1-score	Support
L#	C#	Value	Value	Value	Value
	C#	Value	Value	Value	Value
Macro Averages		**Value**	**Value**	**Value**	

Hamming Loss	Accuracy	Total Time
Value	**Value**	**Value**

Table 11. Experiments presentation template

The values of the dependent variables DV5, DV6, DV7, DV8, DV9 DV10 are bolded since they are the most influencing measures of performance. The experiment results are presented in the implementation section 3.2.2). The interpretation of the values in order to draw conclusions about the performance is discussed in the analysis below.

Experiments Analysis

The analysis of the experiments is performed based on the research sub-questions as an attempt to answer the main research question of the thesis. Two stages of analysis are performed on the experiments results as follows:

1. Analyze the performance of the machine learning classifier in each of the experimental stages.
2. Identify the best algorithm performance in each technique.
3. Analyze the efficiency of each machine learning classification technique based on the best performance identified in step 2.

3.2 Implementation

3.2.1 Application Design Results

The implementation is done by using the artifacts in the sentiment analysis application. The design process for these artifacts is presented in the method section (3.1.1). The result of the artifacts development is presented in the following sections.

Labeled Dataset

The results of labeling the dataset are listed in Table 12 below.

Dataset size	20,000 reviews
Positive reviews	10,008 reviews (50%)
Negative reviews	9,992 reviews (50%)
Comparative reviews	12,145 (61%)
Non-comparative reviews	7,855 (39%)
Aspects-based reviews	12,793 (64%)
Non-aspects-based reviews	7,207 (36%)

Table 12. Labeled dataset implementation results

The labeled dataset is located in the ***datasets*** folder in the repository link for the software application. The folder also includes the source and the development datasets. The link is given in the next section.

Evaluation

For the purpose if evaluating the dataset, binary classification has been performed on the labels to evaluate accuracy of predicting each class. The evaluation was done using MaxEnt algorithm on several iterations.

In the 1st iteration, the dataset was evaluated with 2,000 reviews. The results show that the performance of binary classification was very low. An accuracy of ~60% was recorded for sentiment classification and ~55% for comparison and Aspects-based. This was an indicator that the dataset size was too small for the application.

In the 5th iteration, with a size of 10,000 reviews, the accuracy of the binary classification started to improve. The accuracy was between 70% and 75% for the three labels. In iteration 9 with 18,000 reviews, the accuracy was between 80-85%.

The final 10th iteration was done to get a dataset size of 20,000 reviews. At this size, the binary classification accuracy was ~87% for the sentiment label, ~85% for the comparison label and ~83% for the aspect-based label. The recall for the negative classes, non-comparative and non-aspects-based, was between 78-82%. This is low comparing to the positive classes with values between 83%-87%. However, the labeled dataset was considered as balanced based on the design properties. Therefore, the artifact met the requirements of the application.

Application Software

The repository link https://github.com/yassinha/master-thesis contains all the files of the software developed and used in the application. The module that contains the required functions by the application is called "mlsaa.py". It has the functions to prepare the source and development datasets. Also, the functions to preprocess the labeled dataset, and run the experiments on the classification model. The main functions and their purposes are listed in the table below.

Function name	Description
prepare_development( )	Creates the development dataset from the source.
preprocess( )	Processes the text in the labeled dataset.
get_reviews( )	Gets the reviews text array after preprocessing.
get_labels( )	Gets the labels array that is transformed into a multi-label format.
classify( )	Runs the classification model including the multi-label technique, methods, classifiers and their evaluation.

Table 13. Main functions in the application software

The classify function has the following signature in the software:

```
classify(dataset_name="dataset", ct=1, pt=1, cl=1, …)
```

The parameter ***ct*** is for specifying the multi-label classification technique. The parameter value is the technique number in Table 4. The parameter ***pt*** is for the problem transformation methods which is the method number in Table 5. The parameter ***cl*** is for the machine learning classifier and the value is the classifier number in Table 6. Other parameters are used to define the independent variables in Table 14 and described in the module.

Evaluation

The evaluation of the software developed for the sentiment analysis application was done by manual testing and review during development. Below a summary of the main evaluation steps taken during implementation:

1. The data cleaning was evaluated manually in the dataset while it is being labeled. This is done by making sure all unwanted special characters and extra punctuation have been removed. Also, verifying that contractions were correctly expanded.
2. Multiple iterations have been done on data preparation for the classification model. It was found that some preparation tasks lead to lower performance such as negation handling and stemming, so they were not implemented.
3. The classification model was tested on Iris dataset and on the labeled dataset during labeling.

3.2.2 Experiments Results

The following sections presents the experiment results based on the values of the independent variables and the selected classifier parameters.

Independent Variables

The table below shows the selected values for the dependent variables as fixed values for all the experiments. Also, it gives a rationale of why the value was selected for each variable.

Prefix	Name	Value	Value Rationale
IV1	Dataset Size	20,000 reviews	The size was decided after evaluating the dataset during labeling to achieve a balanced class distribution.
IV2	Training Dataset Size	75% of the original dataset size, i.e. 15,000 reviews	A standard split size adopted in the tools of machine learning.
IV3	Test Dataset Size	25% of the original dataset size, i.e. 5,000 reviews	A standard split size adopted in the tools of machine learning.
IV4	Sampling Method	Standard random sampling method is chosen since the dataset is balanced	Other sampling methods have been tested, such as stratified sampling, but no major influence was observed on the classification results. Therefore, standard sampling method has been selected.
IV5	Cross Validation Folds	10 folds cross validation in order to get a stable measurement	A good number of folds for getting a stable average of the classification performance.
IV6	Features Selection Technique	BoW technique, with word occurrence as binary values 0/1, is chosen to be the most suitable for the problem	Since the purpose of the case study is only an evaluation instead of finding optimum solution, a basic technique was used for features selection. However, it may have a major influence on the classification performance if more advanced feature selection techniques are used.
IV7	Number of Features	5000 features in order to balance between getting stable results and	The number of features was experimented heavily to reach a good number for both the classification performance

		avoiding overfitting the dataset	and in order not to overfit the classifiers with the dataset.
IV8	Minimum Number of n-grams	1, i.e. the features will start from 1 word as a Unigram features in BOW	The words are important in sentiment analysis applications. Therefore, a unigram of words was included in the features.
IV9	Maximum Number of n-grams	3, i.e. the features will include two words as Bigrams and 3 words as Trigrams	Some phrases have good indicators of comparison and product aspects. Therefore, features with up to 3 words are included. It is worth noting that the maximum frequency of words is determined automatically by the number of features. That means it is the best frequency of words and phrases to get the 5000 features.

Table 14. Independent variables values and rationale

Classifiers Parameters

The tables below show the parameters for the classifiers that are used in the experiments. These parameters are fixed among all the experiments. The classifier parameters are usually the default set by the experiment tool. In case if the default value was changed in the experiment, it is shown in bold. It can be referred to the documentation of the class to know more details about the parameters.

CL1 - NB - Naïve Bayes

There are several implementations of Naïve Bayes in Scikit-learn. Multinomial Naïve Bayes, Bernoulli Naïve Bayes and Gaussian Naïve Bayes. They are all based on the same algorithm with adjustments on each one to be suitable for specific types of the data. In the experiments, the class BernoulliNB was used for this classifier, which is more suitable for discrete data and binary values of the features as defined in the independent variable IV6 in Table 14 above.

Classifier Class	sklearn.naive_bayes.BernoulliNB
Classifier Parameters	alpha=1.0, binarize=0.0, fit_prior=True, class_prior=None
Transformation	tfidf=False

Table 15. NB classifier parameters

CL2 - SVM - Support Vector Machine

SVC and LinearSVC are two implementations of SVM in Scikit-learn. The one used is LinearSVC, as it has better support for large scale data.

Classifier Class	sklearn.svm.LinearSVC
Classifier Parameters	penalty='l2', loss='squared_hinge', dual=True, tol=0.0001, C=1.0, multi_class='ovr', fit_intercept=True, intercept_scaling=1, class_weight=None, verbose=0, random_state=None, max_iter=1000
Transformation	tfidf=False

Table 16. SVM classifier parameters

CL3 - MaxEnt - Maximum Entropy

Trials have been performed on the tolerance (tol), however, default value for it gives a very good classification performance. Therefore, Default parameters are used.

Classifier Class	sklearn.linear_model.LogisticRegression
Classifier Parameters	penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='warn', max_iter=100, multi_class='warn', verbose=0, warm_start=False, n_jobs=None
Transformation	tfidf=False

Table 17. MaxEnt classifier parameters

CL4 - kNN - k-Nearest Neighbors

For kNN the number of neighbors to use is adjusted to get a better classification results for the dataset size (IV1) and feature selection technique (IV6) used in the experiments in Table 14.

Classifier Class	sklearn.neighbors.KNeighborsClassifier
Classifier Parameters	**n_neighbors=100**, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs
Transformation	tfidf=True

Table 18. kNN classifier parameters

CL5 - DT - Decision Trees

The maximum depth of the tree was optimized to get a better classification performance.

Classifier Class	sklearn.tree.DecisionTreeClassifier
Classifier Parameters	criterion='gini', splitter='best', **max_depth=15**, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, presort=False
Transformation	tfidf=False

Table 19. DT classifier parameters

CL6 - RF - Random Forest

Two parameters have been optimized for this classifier based on several trials to get a fair classification performance for the problem. The number of trees in the forest (n_estimators) and the maximum depth of the tree (max_depth).

Classifier Class	sklearn.ensemble.RandomForestClassifier
Classifier Parameters	**n_estimators=300**, criterion='gini', **max_depth=15**, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None
Transformation	tfidf=True

Table 20. RF classifier parameters

CL7 - MLkNN - Multi-label k-Nearest Neighbors

The value of k as the number of neighbors of each input instance to take into account has been optimized to achieve fair classification performance. kNN also better works with Tfidf transformation of the input data as noticed in the trials.

Classifier Class	skmultilearn.adapt.MLkNN
Classifier Parameters	**k=20**, s=1.0, ignore_first_neighbours=0
Transformation	tfidf=True

Table 21. MLkNN classifier parameters

Experiment 1

The experiment results below are for applying the binary relevance method of problem transformation technique on Naive Bayes classifier.

EX1	IV10 = CT1, IV11 = PT1, IV12 = CL1 - NB

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.863	0.881	0.872	2493
	C2	0.879	0.860	0.870	2507
L2	C3	0.836	0.802	0.819	3043
	C4	0.710	0.756	0.732	1957
L3	C5	0.854	0.785	0.818	3203
	C6	0.665	0.762	0.710	1797
Macro Averages		**0.801**	**0.808**	**0.803**	

Hamming Loss	Accuracy	Total Time
0.190	**0.535**	**35.423s**

Table 22. Experiment 1 results

Experiment 2

The experiment results below are for applying the binary relevance method of problem transformation technique on SVM classifier.

EX2	IV10 = CT1, IV11 = PT1, IV12 = CL2 - SVM

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.849	0.857	0.853	2495
	C2	0.856	0.848	0.852	2505
L2	C3	0.869	0.867	0.868	3045
	C4	0.794	0.797	0.795	1955
L3	C5	0.849	0.849	0.849	3201
	C6	0.732	0.732	0.732	1799
Macro Averages		**0.825**	**0.825**	**0.825**	

Hamming Loss	Accuracy	Total Time
0.167	**0.581**	**40.166s**

Table 23. Experiment 2 results

Experiment 3

The experiment results below are for applying the binary relevance method of problem transformation technique on Maximum Entropy classifier.

EX3	IV10 = CT1, IV11 = PT1, IV12 = CL3 - MaxEnt

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.881	0.889	0.885	2527
	C2	0.885	0.877	0.881	2473
L2	C3	0.882	0.897	0.889	3042
	C4	0.835	0.813	0.824	1958
L3	C5	0.866	0.872	0.869	3186
	C6	0.772	0.764	0.768	1814
Macro Averages		**0.854**	**0.852**	**0.853**	

Hamming Loss	Accuracy	Total Time
0.140	**0.638**	**32.845s**

Table 24. Experiment 3 results

Experiment 4

The experiment results below are for applying the label powerset method of problem transformation technique on BN classifier.

EX4	IV10 = CT1, IV11 = PT2, IV12 = CL1 - NB

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.860	0.876	0.868	2484
	C2	0.875	0.859	0.867	2516
L2	C3	0.844	0.808	0.825	3044
	C4	0.720	0.767	0.742	1956
L3	C5	0.856	0.775	0.813	3195
	C6	0.659	0.768	0.709	1805
Macro Averages		**0.802**	**0.809**	**0.804**	

Hamming Loss	Accuracy	Total Time
0.189	**0.535**	**27.311s**

Table 25. Experiment 4 results

Experiment 5

The experiment results below are for applying the label powerset method of problem transformation technique on SVM classifier.

EX5	IV10 = CT1, IV11 = PT2, IV12 = CL2 - SVM

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.812	0.814	0.813	2484
	C2	0.816	0.814	0.815	2516
L2	C3	0.813	0.821	0.817	3044
	C4	0.717	0.705	0.711	1956
L3	C5	0.810	0.806	0.808	3195
	C6	0.659	0.664	0.662	1805
Macro Averages		**0.771**	**0.771**	**0.771**	

Hamming Loss	Accuracy	Total Time
0.218	**0.474**	**35.183s**

Table 26. Experiment 5 results

Experiment 6

The experiment results below are for applying the label powerset method of problem transformation technique on MaxEnt classifier.

EX6	IV10 = CT1, IV11 = PT2, IV12 = CL3 - MaxEnt

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.841	0.847	0.844	2484
	C2	0.848	0.842	0.845	2516
L2	C3	0.833	0.861	0.846	3044
	C4	0.771	0.731	0.750	1956
L3	C5	0.820	0.840	0.830	3195
	C6	0.704	0.673	0.688	1805
Macro Averages		**0.803**	**0.799**	**0.801**	

Hamming Loss	Accuracy	Total Time
0.189	**0.529**	**31.320s**

Table 27. Experiment 6 results

Experiment 7

The experiment results below are for applying the classifier chain method of problem transformation technique on NB classifier.

EX7	IV10 = CT1, IV11 = PT3, IV12 = CL1 - NB

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.861	0.883	0.872	2484
	C2	0.882	0.860	0.871	2516
L2	C3	0.834	0.799	0.816	3044
	C4	0.706	0.752	0.728	1956
L3	C5	0.854	0.785	0.818	3195
	C6	0.667	0.762	0.711	1805
Macro Averages		**0.801**	**0.807**	**0.803**	

Hamming Loss	Accuracy	Total Time
0.191	**0.533**	**56.891s**

Table 28. Experiment 7 results

Experiment 8

The experiment results below are for applying the classifier chain method of problem transformation technique on SVM classifier.

EX8	IV10 = CT1, IV11 = PT3, IV12 = CL2 - SVM

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.851	0.859	0.855	2484
	C2	0.860	0.851	0.856	2516
L2	C3	0.873	0.869	0.871	3044
	C4	0.797	0.802	0.800	1956
L3	C5	0.851	0.850	0.850	3195
	C6	0.735	0.737	0.736	1805
Macro Averages		**0.828**	**0.828**	**0.828**	

Hamming Loss	Accuracy	Total Time
0.164	**0.586**	**43.484s**

Table 29. Experiment 8 results

Experiment 9

The experiment results below are for applying the classifier chain method of problem transformation technique on MaxEnt classifier.

EX9	IV10 = CT1, IV11 = PT3, IV12 = CL3 - MaxEnt

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.876	0.883	0.880	2484
	C2	0.884	0.877	0.880	2516
L2	C3	0.886	0.894	0.890	3044
	C4	0.833	0.822	0.827	1956
L3	C5	0.868	0.871	0.869	3195
	C6	0.770	0.765	0.768	1805
Macro Averages		**0.853**	**0.852**	**0.852**	

Hamming Loss	Accuracy	Total Time
0.140	**0.636**	**39.770s**

Table 30. Experiment 9 results

Experiment 10

The experiment results below are for applying problem transformation technique on kNN classifier (using out of the box implementation of Scikit-learn).

EX10	IV10 = CT1, IV11 = N/A, IV12 = CL4 - kNN

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.823	0.887	0.854	2484
	C2	0.889	0.791	0.837	2516
L2	C3	0.667	0.957	0.786	3044
	C4	0.806	0.226	0.353	1956
L3	C5	0.706	0.924	0.800	3195
	C6	0.708	0.291	0.412	1805
Macro Averages		**0.767**	**0.679**	**0.674**	

Hamming Loss	Accuracy	Total Time
0.257	**0.406**	**29.789s**

Table 31. Experiment 10 results

Experiment 11

The experiment results below are for applying problem transformation technique on DT classifier (using out of the box implementation of Scikit-learn).

EX11	IV10 = CT1, IV11 = N/A, IV12 = CL5 - DT

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.746	0.761	0.752	2484
	C2	0.760	0.744	0.751	2516
L2	C3	0.823	0.782	0.802	3044
	C4	0.689	0.737	0.712	1956
L3	C5	0.794	0.704	0.746	3195
	C6	0.566	0.674	0.615	1805
Macro Averages		**0.730**	**0.734**	**0.730**	

Hamming Loss	Accuracy	Total Time
0.262	**0.378**	**40.183s**

Table 32. Experiment 11 results

Experiment 12

The experiment results below are for applying problem transformation technique on RF classifier (using out of the box implementation of Scikit-learn).

EX12	IV10 = CT1, IV11 = N/A, IV12 = CL6 - RF

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.829	0.831	0.830	2484
	C2	0.831	0.834	0.832	2516
L2	C3	0.744	0.981	0.847	3044
	C4	0.941	0.475	0.631	1956
L3	C5	0.688	0.973	0.806	3195
	C6	0.828	0.214	0.340	1805
Macro Averages		**0.810**	**0.718**	**0.714**	

Hamming Loss	Accuracy	Total Time
0.228	**0.436**	**75.396s**

Table 33. Experiment 12 results

Experiment 13

The experiment results below are for applying algorithm adaption technique on MLkNN classifier (using out of the box implementation of Scikit-learn).

EX13	IV10 = CT2, IV11 = N/A, IV12 = CL7 - MLkNN

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.857	0.869	0.863	2547
	C2	0.862	0.850	0.856	2453
L2	C3	0.787	0.778	0.782	3064
	C4	0.655	0.666	0.660	1936
L3	C5	0.780	0.797	0.788	3176
	C6	0.632	0.607	0.620	1824
Macro Averages		**0.762**	**0.761**	**0.762**	

Hamming Loss	Accuracy	Total Time
0.226	**0.470**	**103.936s**

Table 34. Experiment 13 results

Experiment 14

The experiment results below are for applying ensemble (RAKEL) technique on NB classifier.

EX14	IV10 = CT3, IV11 = N/A, IV12 = CL1 - NB

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.868	0.872	0.870	2484
	C2	0.873	0.869	0.871	2516
L2	C3	0.841	0.808	0.825	3044
	C4	0.717	0.762	0.739	1956
L3	C5	0.858	0.785	0.820	3195
	C6	0.669	0.770	0.716	1805
Macro Averages		**0.804**	**0.811**	**0.807**	

Hamming Loss	Accuracy	Total Time
0.187	**0.513**	**25.199s**

Table 35. Experiment 14 results

Experiment 15

The experiment results below are for applying ensemble (RAKEL) technique on SVM classifier.

EX15	IV10 = CT3, IV11 = N/A, IV12 = CL2 - SVM

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.829	0.837	0.833	2484
	C2	0.833	0.823	0.828	2516
L2	C3	0.832	0.833	0.832	3044
	C4	0.743	0.736	0.739	1956
L3	C5	0.821	0.821	0.821	3195
	C6	0.683	0.683	0.683	1805
Macro Averages		**0.790**	**0.789**	**0.789**	

Hamming Loss	Accuracy	Total Time
0.201	**0.461**	**37.633s**

Table 36. Experiment 15 results

Experiment 16

The experiment results below are for applying ensemble (RAKEL) technique on MaxEnt classifier.

EX16	IV10 = CT3, IV11 = N/A, IV12 = CL3 - MaxEnt

Label	Class	Precision	Recall	F1-score	Support
L1	C1	0.855	0.858	0.857	2484
	C2	0.859	0.857	0.858	2516
L2	C3	0.845	0.868	0.856	3044
	C4	0.781	0.752	0.766	1956
L3	C5	0.833	0.850	0.842	3195
	C6	0.722	0.696	0.709	1805
Macro Averages		**0.816**	**0.813**	**0.815**	

Hamming Loss	Accuracy	Total Time
0.175	**0.522**	**31.328s**

Table 37. Experiment 16 results

4 Findings and Analysis

This section represents the answers of the main research question "How good is using multi-label classification techniques in identifying comparative opinions based on product aspects and their sentiment classification?".

4.1 Performance of Machine Learning Algorithms

The research sub-question that is answered in this section is "What is the performance of machine learning algorithms when used in each multi-label classification techniques?". The experiment stages in section 3.1.9 are used as a structure for presenting the finding in the following sections.

4.1.1 Performance of the Algorithms using Multi-label Problem Transformation Technique

Problem transformation technique has three methods. The performance of three machine learning algorithms is measured in each method. The performance measurement is expressed by the 6 dependent variables of Macro Precision, Macro Recall, Macro F1-Score, Hamming Loss, Accuracy and Total Time. The highest values are bolded in the presentation tables.

Binary Relevance

The table below shows the performance of the algorithms when used with Binary Relevance method.

Classifier	Macro Precision	Macro Recall	Macro F1-score	Hamming Loss	Accuracy	Total Time
NB	0.801	0.808	0.803	0.190	0.535	35.423s
SVM	0.825	0.825	0.825	0.167	0.581	40.166s
MaxEnt	**0.854**	**0.852**	**0.853**	**0.140**	**0.638**	**32.845s**

Table 38. Algorithms performance with binary relevance method

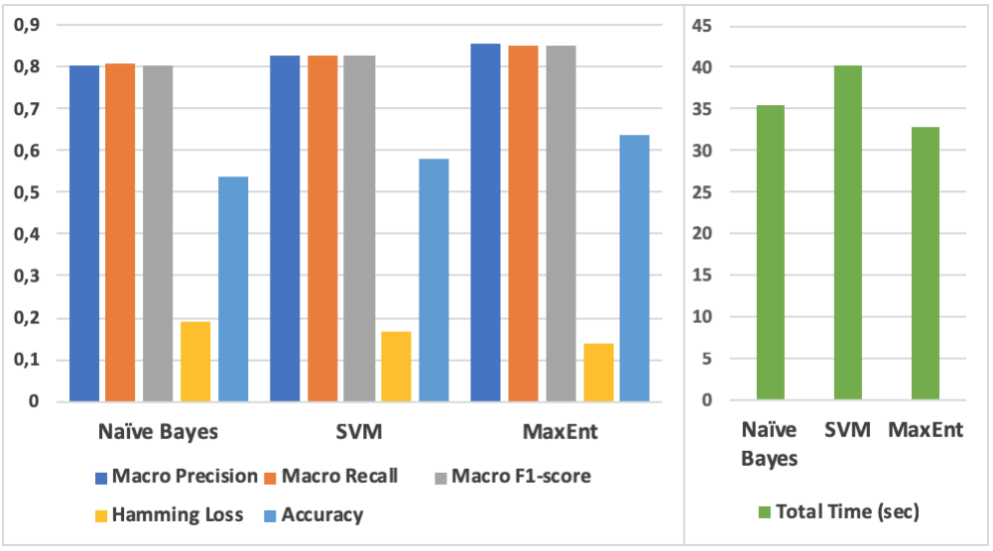

Figure 10. Algorithms performance with binary relevance method.

Label Powerset

The table below shows the performance of the algorithms when used with Label Powerset method.

Classifier	Macro Precision	Macro Recall	Macro F1-score	Hamming Loss	Accuracy	Total Time
NB	0.802	**0.809**	**0.804**	**0.189**	**0.535**	**27.311s**
SVM	0.771	0.771	0.771	0.218	0.474	35.183s
MaxEnt	**0.803**	0.799	0.801	0.189	0.529	31.320s

Table 39. Algorithms performance with label powerset method

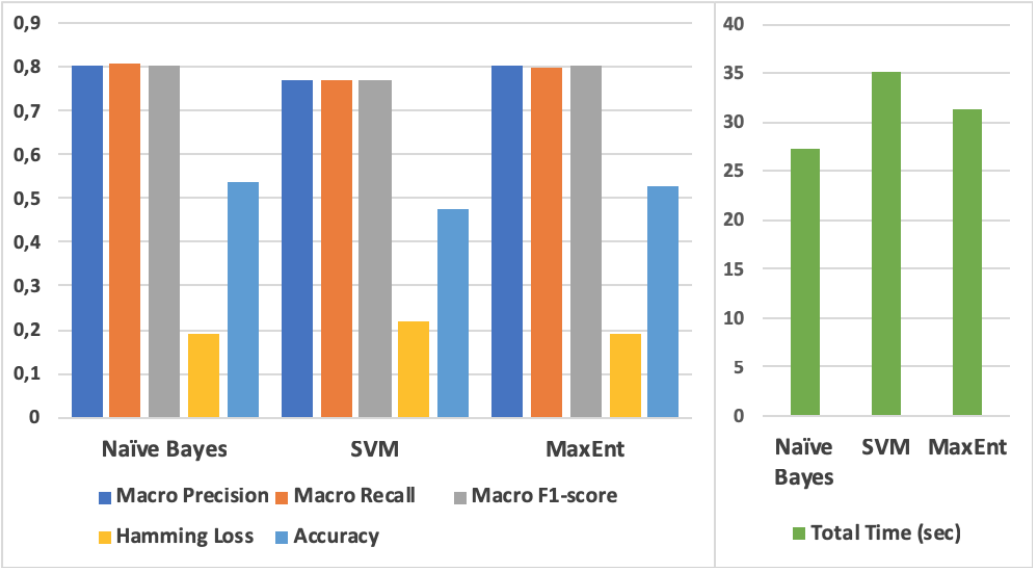

Figure 11. Algorithms performance with label powerset method.

Classifier Chain

The table below shows the performance of the algorithms when used with Label Powerset method.

Classifier	Macro Precision	Macro Recall	Macro F1-score	Hamming Loss	Accuracy	Total Time
NB	0.801	0.807	0.803	0.191	0.533	56.891s
SVM	0.828	0.828	0.828	0.164	0.586	43.484s
MaxEnt	**0.853**	**0.852**	**0.852**	**0.140**	**0.636**	**39.770s**

Table 40. Algorithms performance with label classifier chain method

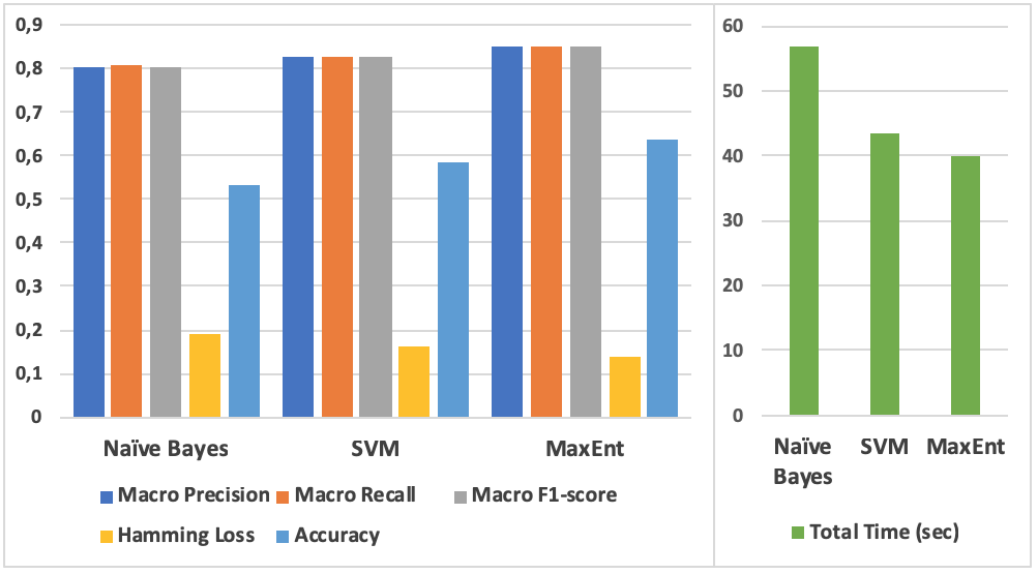

Figure 12. Algorithms performance with classifier chain method.

Direct Problem Transformation

Table 41 below shows the performance of the algorithms when used with Label Powerset method.

Classifier	Macro Precision	Macro Recall	Macro F1-score	Hamming Loss	Accuracy	Total Time
kNN	0.767	0.679	0.674	0.257	0.406	**29.789s**
DT	0.730	**0.734**	**0.730**	0.262	0.378	40.183s
RF	**0.810**	0.718	0.714	**0.228**	**0.436**	75.396s

Table 41. Algorithms performance with direct problem transformation method

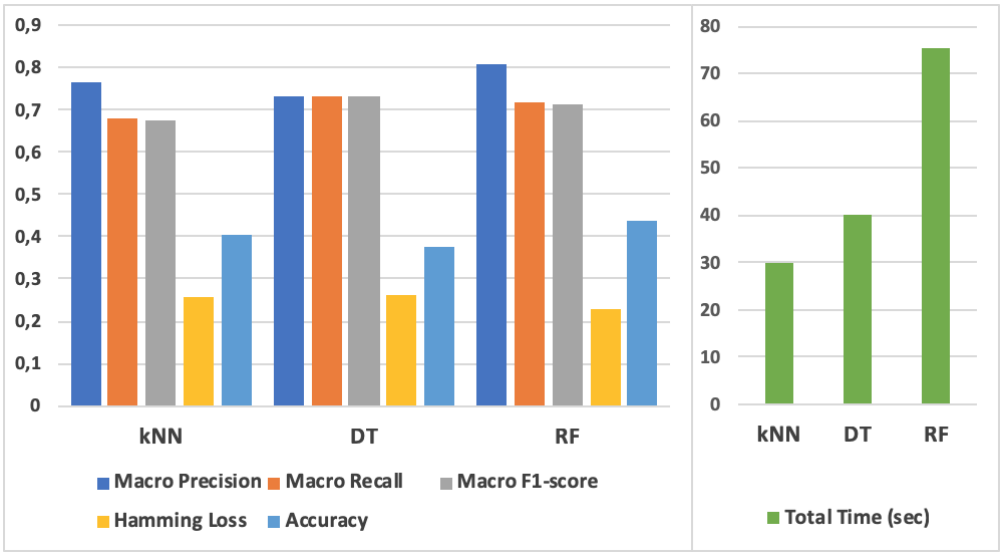

Figure 13. Algorithms performance with direct problem transformation method.

It can be observed in the problem transformation method technique that MaxEnt classifier is with the classifier with highest performance in two of the problem transformation methods. Naïve Bayes scored the top performance in the Label Powerset technique. However, the performance of applying problem transformation method on kNN, DT and RF directly with their default multi-label implementation has lower performance comparing to the other problem transformation methods.

Top performance among the three methods of the problem transformation technique is for MaxEnt classifier using Binary Relevance.

Classifier	Macro Precision	Macro Recall	Macro F1-score	Hamming Loss	Accuracy	Total Time
Binary Relevance of MaxEnt	**0.854**	**0.852**	**0.853**	**0.140**	**0.638**	**32.845s**

Table 42. Top algorithm performance in problem transformation technique

4.1.2 Performance of the Algorithms using Multi-label Algorithm Adaptation Technique

Measuring the algorithms performance in this technique depends on the availability of adapted algorithms in terms of the practical implementation of that algorithm in machine learning tools. Therefore, the performance of MLkNN algorithm was only measured as it is implemented in the machine learning tools used in the experiments.

MLkNN

Table 43 below shows the performance of the MLkNN algorithm:

Classifier	Macro Precision	Macro Recall	Macro F1-score	Hamming Loss	Accuracy	Total Time
MLkNN	0.762	0.761	0.762	0.226	0.470	103.936s

Table 43. Algorithms performance with label powerset method

It can be observed in Table 43 that the performance of MLkNN improved compared to kNN. The Macro F1-score is increased by 10%. However, the total time was doubled three times comparing to kNN.

4.1.3 Performance of the algorithms using Multi-label Ensemble Technique

Multi-label Ensemble technique is applied using RAKEL method. The performance of three machine learning algorithms is evaluated this method. The performance is expressed by the 6 dependent variables of Macro Precision, Macro Recall, Macro F1-Score, Hamming Loss, Accuracy and Total Time. The highest values are bolded in the presentation tables.

RAKEL

Table 44 below shows the performance of the algorithms when used with Ensemble (RAKEL) multi-label technique.

Classifier	Macro Precision	Macro Recall	Macro F1-score	Hamming Loss	Accuracy	Total Time
NB	0.804	0.811	0.807	0.187	0.513	**25.199s**
SVM	0.790	0.789	0.789	0.201	0.461	37.633s
MaxEnt	**0.816**	**0.813**	**0.815**	**0.175**	**0.522**	31.328s

Table 44. Algorithms performance with Ensemble (RAKEL) technique

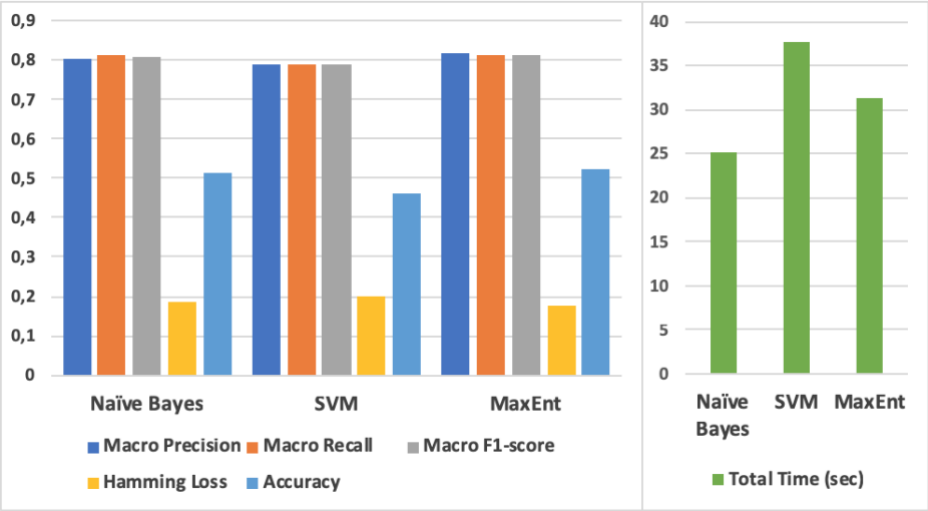

Figure 14. Algorithms performance with Ensemble (RAKEL) technique.

It can be observed that ensemble technique using RAKEL has better performance than some of the problem transformation methods such as label powerset. MaxEnt algorithms scores the top performance using RAKEL with Macro averages of F1-score 81.5% and average total time.

4.2 Efficiency of Multi-label Classification Techniques

The research sub-question that is answered in this section is "What is the efficiency of each multi-label classification technique in terms of the best performance of the machine learning algorithms used?". The performance measurements in section 4.1 are used as the basis of the analysis.

In order to describe the efficiency depending on the performances, a scale is developed based on the Macro F1-score metric:

a. Excellent**:** score greater than 90%
b. Very good: score between 80% - 90%
c. Good: score between 70 - 80%
d. Poor: score between 60 - 70%
e. Very Poor: score less than 60%

4.2.1 Problem Transformation Technique

Performance score for the top classifiers in each method are listed in Table 45 below.

Method	Classifier	Macro F1-score	Hamming Loss	Accuracy	Total Time
Binary Relevance	**MaxEnt**	**0.853**	**0.140**	**0.638**	**32.845s**
Label Powerset	NB	0.804	0.189	0.535	27.311s
Classifier Chain	MaxEnt	0.852	0.140	0.636	39.770s
Direct Problem Transformation	RF	0.714	0.228	0.436	75.396s

Table 45. Algorithm top scores for problem transformation technique

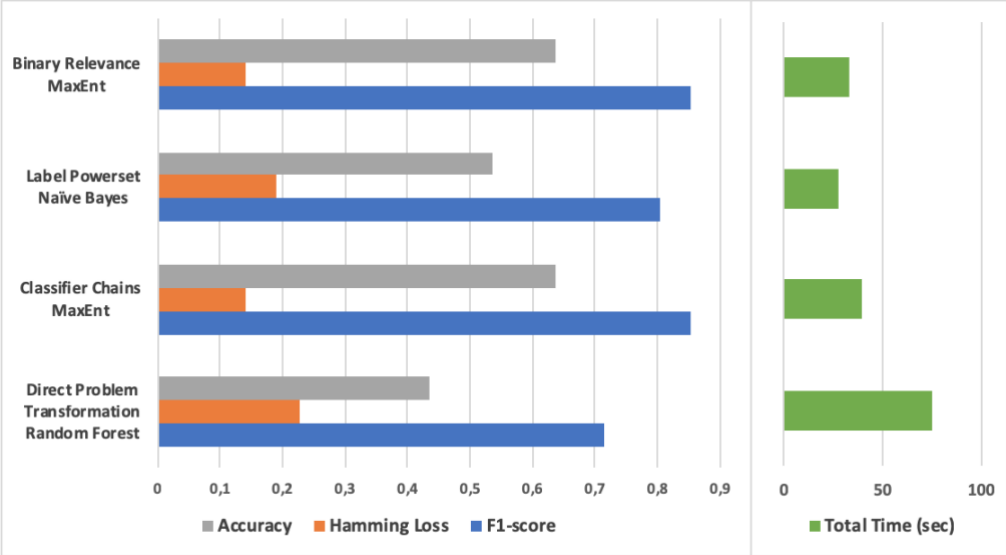

Figure 15. Algorithm top scores for problem transformation technique.

The efficiency of the technique is very good with top Macro F1-score of 85.3%.

4.2.2 Algorithm Adaptation Technique

Performance score for the MLkNN classifiers is given in Table 46 below.

Method	Classifier	Macro F1-score	Hamming Loss	Accuracy	Total Time
Algorithm Adaptation	**MLkNN**	**0.762**	**0.226**	**0.470**	**103.936s**

Table 46. MLkNN scores for algorithm adaptation technique

The efficiency of the technique is good with Macro F1-score of 76.2%.

4.2.3 Ensemble (RAKEL) Technique

Performance score for the classifiers using ensemble RAKEL method are listed in Table 47 below.

Classifier	Macro F1-score	Hamming Loss	Accuracy	Total Time
NB	0.807	0.187	0.513	**25.199s**
SVM	0.789	0.201	0.461	37.633s
MaxEnt	**0.815**	**0.175**	**0.522**	31.328s

Table 47. Algorithm scores for ensemble (RAKEL) technique

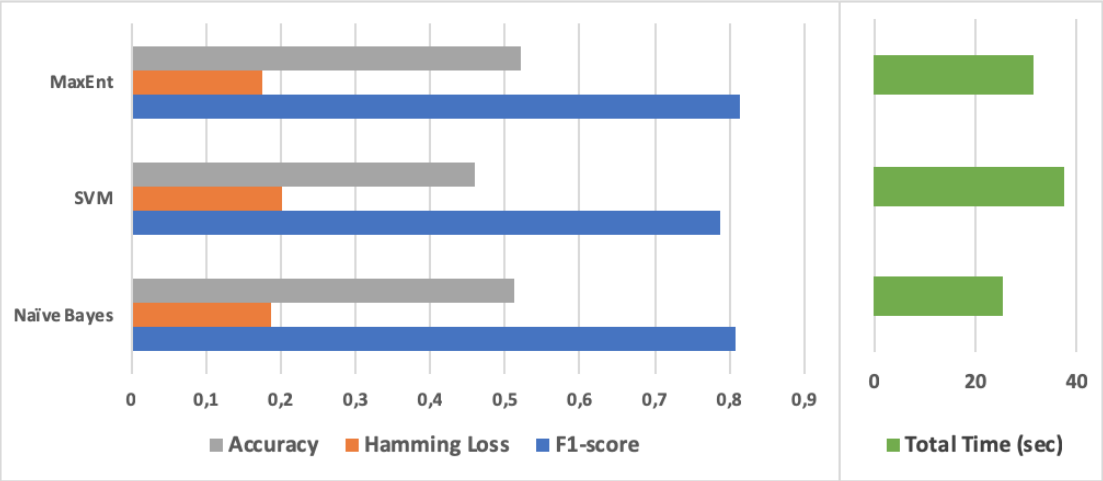

Figure 16. Algorithm scores for ensemble (RAKEL) technique.

The efficiency of the technique is very good with top Macro F1-score of 81.3%.

4.3 Effectiveness of Multi-label Classification Techniques

The research sub-question that is answered in this section is "What is the effectiveness of multi-label machine learning classification techniques?" The efficiency findings in previous analysis, section 4.2, are used as the basis of the effectivity analysis. There answer to the question is presented as effective or not effective for each of the techniques and on average for the multi-label techniques. Table below shows the efficiency scale of each technique and its effectiveness. The total time of training and testing is not included in the analysis, since effectiveness of classification is not influenced by time as long as it is completed.

The efficiency scale, which is based on Macro F1-score, is considered poor or very poor when the score is less than 70%. Poor efficiency is considered ineffective. Therefore, the effectiveness decision is based on the following scale:

a. Effective**:** when efficiency is good, very good, or excellent (score > 70%).
b. Ineffective: when efficiency is poor or very poor (score < 70%).

Technique	Method	Classifier	Efficiency	Effective
Problem Transformation	**Binary Relevance**	**MaxEnt**	**Very Good**	**Yes**
Algorithm Adaptation	Adapt kNN	MLkNN	Good	Yes
Ensemble	RAKEL	MaxEnt	Very Good	Yes

Table 48. Effectiveness of each multi-label classification technique

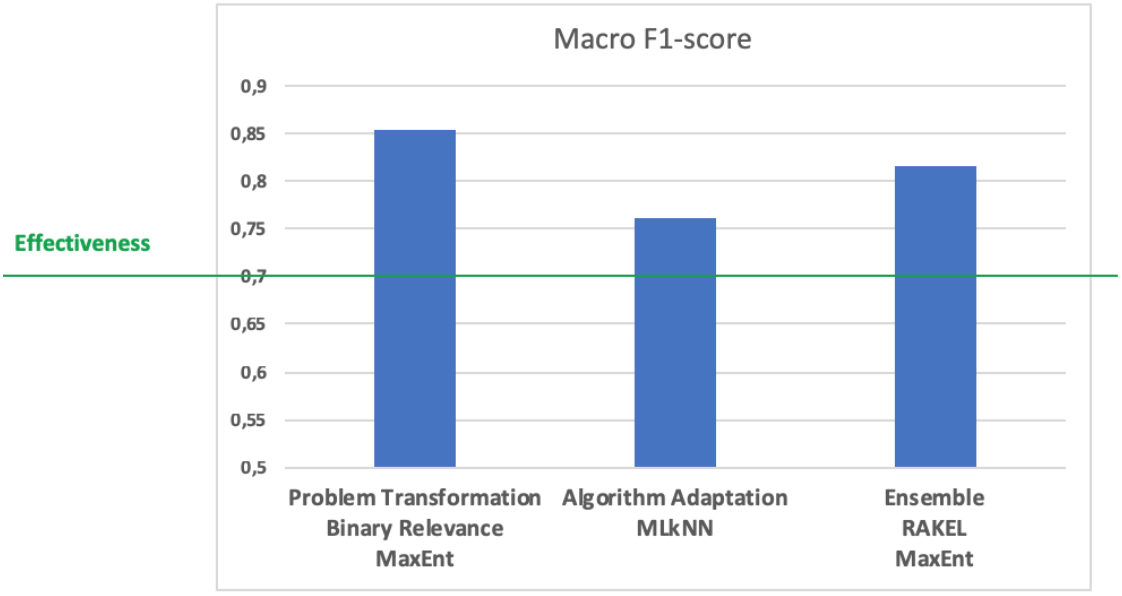

Figure 17. Scores and effectiveness of each multi-label classification technique.

It can be observed in Table 48 and Figure 17 that all multi-label classification techniques evaluated in this case study provide an effective solution to the sentiment analysis application under study.

5 Discussion and Conclusion

5.1 Discussion of Method

The purpose of this research is to evaluate the effectiveness and efficiency of multi-label classification techniques in identifying comparative opinions based on product aspects. To perform the evaluation, a dataset of online customer reviews was prepared. Then, a classification model was developed in a suitable way for multi-label classification techniques. Finally, empirical experiments were conducted on the classification model by using different machine learning algorithms to evaluate the techniques.

Design and creation research method was used to develop the artifacts that are needed for the evaluation in a structured way. The artifacts are the labeled dataset and the application software that includes the classification model and the functions to execute the empirical experiments.

The process which is followed to design and develop the artifacts starts with problem definition. Then, the steps needed for designing, developing and implementing the artifacts. Finally, evaluation is performed on the artifacts. The process itself is simple, but it is necessary for structuring the research work in this part of the case study. It was very effective and focused on the artifacts while following the design and development steps.

For the empirical experiments, a structured process was followed to define the scope, planning, implementation, presentation and analysis of the experiments. First, the scope was defined based on the research purpose. Second, the dependent and independent variables in addition to the instruments and tools were defined. Third, the method was explained for implementation, representation and analysis of the experiments. The process was very useful in setting a structure for the empirical experiments. Furthermore, since comparison is not needed, the analysis of the findings did not require advanced techniques.

The two parts of the research were performed as a case study of the sentiment analysis application on online customer reviews. The case study research method represents a good framework for evaluating a specific model in a defined context, especially in the software engineering field. Also, they are more suitable when the findings are not necessarily to be generalized on all similar models.

5.2 Discussion of Findings

The purpose of the case study is the evaluation of the effectiveness and efficiency of multi-label classification techniques in the sentiment analysis application under study. Therefore, the findings of the case study are the evaluation results represented by the efficiency and effectiveness of each technique. The performance of machine learning algorithms in each technique is used to evaluate the efficiency of it. Then, the effectiveness is decided based on the efficiency. The findings are structured to answer the main research question and sub-questions.

5.2.1 What is the performance of machine learning algorithms when used in each multi-label classification techniques?

The multi-label classification techniques are applied to several classifiers in the experiments. The first step toward evaluating the technique is measuring the performance of the classifiers. The performance measurement is represented by the evaluation metrics of Macro Precision (MP), Macro Recall (MR), Macro F1-score (MF1), Hamming Loss (HL), Accuracy (AC) and Total Time for training and testing the classifier.

MF1 metric is the harmonic mean of both MP and MR, so there is already a correlation between these three values. Furthermore, there was a strong correlation between MF1, HL and AC. When MF1 value increases, the AC value increases and the HL value decreases. Therefore, the performance measurements were easier to analyze in terms of finding best performance.

MaxEnt classifier scored the best performance in the problem transformation technique using the binary relevance method. The classifier scored a MF1 of 85.3%, an AC of 63.8% and a HL of 14%. Also, MaxEnt scored top performance in the classifier chain method with 85.2% MF1, 63.6% AC and 14% HL. Naïve Bayes scored the top performance using Label powerset of the same technique with 80.4% MF1, 53.5% AC and 18.9% HL. However, MaxEnt performance was very close with 80.1% MF1-score, 52.9% accuracy and same HL.

It can be observed clearly that when the MF1 value changes, other values change in correlation. Therefore, it was easily concluded that MaxEnt has the best performance using problem transformation technique. The total time is also the lowest with 32.8 seconds in binary relevance and 39.7 seconds in classifier chain. It was also observed that the total time in classifier chain method increases by 25% of total time for all classifiers comparing to other methods.

The performance was low for kNN, DT and RF classifiers with their multi-label implementation in Scikit-learn tool. The lowest was kNN with 67.4% MF1, 40.6% AC and 25.7% HL. The RF classifier performed better with 71.4% MF1, 43.6% AC and 22.8% HL. However, the total time of RF was almost double that of kNN with 75 seconds. The low performance of these classifiers was expected since they are not considered very good classifiers for text classification problems in general.

In the algorithm adaptation technique, MLkNN has scored an average performance with 76.2% MF1, 47% AC and 22.6% HL. The performance was better than the one for kNN in problem transformation technique as MF1 was improved by 10%.

For Ensemble (RAKEL), MaxEnt classifier has proved to be a good classifier in this technique as well. It has the top performance with 81.5% MF1, 53.2% AC and 17.5% HL. Furthermore, the total time taken by all the classifier in Ensemble (RAKEL) technique was the low comparing to other techniques with an average of 31 seconds. This is against the expectations and indicates that this technique is efficient in terms of the time.

5.2.2 What is the efficiency of each multi-label classification technique in terms of the best performance of the machine learning algorithms used?

The efficiency of the multi-label classification techniques is evaluated based on the best performance for the machine learning classifiers used in the technique. Since there was a strong correlation between Macro F1-score (MF1), Accuracy (AC) and Hamming Loss (HL) metrics, the MF1 metric was used for evaluating the efficiency. A scale was suggested in this research for evaluating the efficiency. It is considered poor if MF1 is less than 70%, good if MF1 is between 60% and 70%, very good if MF1 is between 70% and 80% and excellent if MF1 is greater than 90%.

For the problem transformation technique, the best performance was measured for MaxEnt classifier with 85.3% MF1. This means that the efficiency of the technique is very good. In general, most of the classifiers have a very good efficiency in the technique. The SVM classifier using label powerset method has a good efficiency with 77.1% MF1. The only poor efficiency was observed for kNN with 67.9% MF1.

For the algorithm adaptation technique, only the performance of MLkNN classifier was measured with 76.2% MF1. This means that the efficiency of the technique using MLkNN is good. It was noticed that the total time taken by MLkNN classifier in training and testing is very high with 103.9 seconds.

For the ensemble technique using RAKEL method, the performance of MaxEnt classifier was the best with 81.5% MF1. This means that the efficiency of the technique is very good. Naïve Bayes classifier using RAKEL scored a very good efficiency as well with 80.7% MF1. However, the performance of SVM classifier was lower with 78.9% MF1.

The most efficient technique was the problem transformation using binary relevance method with MaxEnt classifier. The MF1 of the classifier was 85.3%. None of technique was able to achieve an excellent efficiency with MF1 more than 90%. It may need further improvements in the classification model such as implementing a better features selection technique.

5.2.3 What is the effectiveness of multi-label machine learning classification techniques?

The effectiveness of the techniques is evaluated based on the efficiency evaluation. Therefore, if the efficiency of a technique is not poor, then the technique is considered effective for the sentiment analysis application. In general, none of the evaluated multi-label techniques has a poor efficiency, so all the evaluated techniques are considered effective.

The most effective technique is the problem transformation using both binary relevance and classifier chain methods. These methods have a very good efficiency for all the classifiers that were used in the methods.

The most effective classifier for the sentiment analysis application is MaxEnt when used with binary relevance method within the problem transformation technique.

5.3 Conclusion

Mining comparative opinions based on product aspects is a sentiment analysis application that is getting attention by researchers and businesses. Different approaches and techniques have been proposed for this problem. With the continuous development of machine learning, new techniques have been introduced such as multi-label learning techniques. This thesis proposed an evaluation of using multi-label machine learning techniques in this sentiment analysis application.

The application has been designed and developed in a suitable way for multi-label learning. The first step was preparing a labeled dataset. There is a lack of free data sources for this specific application. Therefore, manual labeling was needed. A dataset of Amazon online customer reviews was used to create a labeled dataset of 20,000 reviews. The reviews were labeled manually for comparative and aspects-based. This process was an exhausting task that took months to be completed. Therefore, the dataset can be added to the contributions of this research.

The labeled dataset of online reviews was used to develop the classification model for the evaluation of multi-label classification techniques. The reviews were preprocessed for features selection using BoW technique. The labels were binarized in a suitable way for multi-label classification. Then, the classification model was developed by using Python and Scikit-learn tool. The tool has all the functions for using the classifiers with the multi-label techniques. The implementation was rather easy due to the convenient way of using the functions in the tool. The tool may not be ready for large scale use in business applications. However, it is sufficient for the evaluation purpose on this research.

The multi-label techniques were evaluated in terms of their effectiveness and efficiency in addressing the sentiment analysis application. The efficiency of the techniques was determined based on the performance of different machine learning classifiers. Whereas, the effectiveness was determined based on the efficiency. All the techniques were found to be effective with addressing the problem. The efficiency was very good for Problem Transformation and Ensemble (RAKEL) techniques. The best performance was scored by MaxEnt classifier with a Macro F1-score of 85.3% by using Binary Relevance method.

For future work, the labeled dataset created in this research can be refined and standardized as a corpus for such applications. Also, it is expected that improving the features selection technique will have a major influence on the performance of the classifiers. With a more standardized dataset and better features engineering, comparative analysis can be performed on the techniques to find the optimum solution for the sentiment analysis application.

6 References

Abramyk, H. (2017, February 27). *Top 10 Customer Review Websites*. Retrieved Augut 2018, from Vendasta: https://www.vendasta.com/blog/top-10-customer-review-websites

Aggarwal, C. C., Zhai, C., & editors. (2012). *Mining text data.* Springer Science & Business Media.

Ahmad, M., Aftab, S., Ali, I., & Hameed, N. (2017). Hybrid Tools and Techniques for Sentiment Analysis: A Review. *Int. J. Multidiscip. Sci. Eng, 8*(3), 28-33.

Alpaydin, E. (2014). *Introduction to machine learning. 2nd ed.* s.l.:MIT Press.

Appel, O., Chiclana, F., & Carter, J. (2015). Main Concepts, State of the Art and Future Research Questions in Sentiment Analysis. *Acta Polytechnica Hungarica, 12*(3), 87-108.

Basili, V. R., & Rombach., H. D. (1988). The TAME project: Towards improvement-oriented software environments. *IEEE Transactions on software engineering, 14*(6), 758-773.

Benamara, F., Cesarano, C., Picariello, A., Reforgiato, D., & Subrahmanian, V. S. (2007). Sentiment analysis: Adjectives and adverbs are better than adjectives alone. *In Proceedings of the International Conference on Weblogs and Social Media (ICWSM).*

Bhadane, C., Dalal, H., & Doshi, H. (2015). Sentiment analysis: Measuring opinions. *Procedia Computer Science. 45*, pp. 808-814. Elsevier.

Bird, S., Loper, E., & Klein, E. (2009). *Natural Language Processing with Python.* O'Reilly Media Inc.

Chowdhury, G. G. (2003). Natural Language Processing. *Annual review of information science and technology, 37*(1), 51-89.

Cui, H., Mittal, V., & Datar, M. (2007). Comparative Experiments on Sentiment Classification for Online Product Reviews. *AAAI, 6*, 1265-1270.

Destercke, S. (2014). Multilabel prediction with probability sets: the hamming loss case. *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems* (pp. 496-505). Cham: Springer.

Dubey, G., Rana, A., & Ranjan, J. (2017). Fine-grained opinion mining of product review using sentiment and semantic orientation. *International Journal of Business Information Systems, 25*(1), 1-17.

Geurts, P., Ernst., D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning, 63*(1), 3-42.

Ghag, K., & Shah, K. (2013). Comparative analysis of the techniques for Sentiment Analysis. *International Conference on Advances in Technology and Engineering (ICATE)* (pp. 1-7). IEEE.

Gilbert, C. H. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. *Proceedings of the Eighth AAAI International Conference on Weblogs and Social Media* (pp. 216-225). AAAI.

He, R., & McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. *Proceedings of the 25th international conference on world wide web* (pp. 507-517). International World Wide Web Conferences Steering Committee.

Horrigan, J. (2008). *Online shopping: Internet users like the convenience but worry about the security of their financial information.* Washington, DC: Pew Internet & American Life Project.

Hu, M., & Liu, B. (2004). Mining and Summarizing Customer Reviews. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 168-177). ACM.

Jian, J., Ji, P., & Yan, S. (2017). Comparison of series products from customer online concerns for competitive intelligence. *Journal of Ambient Intelligence and Humanized Computing*, 1-16.

Jindal, N., & Liu, B. (2006). Identifying Comparative Sentences in Text Documents. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 244-251). ACM.

Kaggle. (2017, December 19). *Toxic Comment Classification Challenge*. Retrieved November 2018, from Kaggle: https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

Khan, A. U., Khan, M., & Khan, M. B. (2016). Naïve Multi-label classification of YouTube comments using comparative opinion mining. *Procedia Computer Science. 82*, pp. 57-64. Elsevier.

Lima, A. C., & Castro, L. N. (2014). A multi-label, semi-supervised classification approach applied to personality prediction in social media. *Neural Networks, 58*, 122-130.

Liu, B. (2012). Sentiment Analysis and Opinion Mining. *Synthesis lectures on human language technologies, 5*(1), 1-167.

Liu, S. M., & Chen, J.-H. (2015). A multi-label classification based approach for sentiment classification. *Expert Systems with Applications, 42*(3), 1083-1093.

Mäntylä, M. V., Graziotin, D., & Kuutila, M. (2018). The evolution of sentiment analysis—A review of research topics, venues, and top cited papers. *Computer Science Review 27*, 16-32.

Madjarov, G., Kocev, D., Gjorgjevikj, D., & Džeroski, S. (2012). An extensive experimental comparison of methods for multi-label learning. *Pattern recognition, 45*(9), 3084-3104.

McAuley, J., & Yang, A. (2016). Addressing Complex and Subjective Product-Related Queries with Customer Reviews. *Proceedings of the 25th International Conference on World Wide Web* (pp. 625-635). International World Wide Web Conferences Steering Committee.

McCallum, A. (1999). Multi-label text classification with a mixture model trained by EM. *AAAI workshop on Text Learning*.

Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal, 5*(4), 1093-1113.

Miroslav, K. (2015). *An Introduction to Machine Learning.* s.l.:Springer International Publishing.

Mount, J. (2011, September 23). *The equivalence of logistic regression and maximum entropy models*. Retrieved October 2018, from Win-Vector: http://www.win-vector.com/dfiles/LogisticRegressionMaxEnt.pdf

Mubarok, M. S., Adiwijaya, & Aldhi, M. D. (2017). Aspect-based Sentiment Analysis to Review Products Using Naïve Bayes. *AIP Conference Proceedings. 1867*, p. 020060. AIP Publishing.

Murthy, G., & Liu, B. (2008). Mining Opinions in Comparative Sentences. *Proceedings of the 22nd International Conference on Computational Linguistics. 1*, pp. 241-248. Association for Computational Linguistics.

NLTK. (2018, October 28). *NLTK Project*. Retrieved October 2018, from NLTK Project: http://www.nltk.org/

Oates, B. J. (2006). *Researching information systems and computing.* London: SAGE.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval 2.1, 2*, 1-135.

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment Classification using Machine Learning Techniques. *Proceedings of the ACL-02 conference on Empirical methods in natural language processing. 10*, pp. 79-86. Association for Computational Linguistics.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Brucher, M. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research, 12*(Oct), 2825-2830.

Peffers, K., Tuunanen, T., Gengler, C. E., Rossi, M., Hui, W., Virtanen, V., & Bragge, J. (2006). The design science research process: a model for producing and presenting information systems research. *Proceedings of the first international conference on design science research in information systems and technology* (pp. 83-106). DESRIST 2006.

PyPI. (2018, November 28). *pycontractions*. Retrieved November 2018, from PyPI: https://pypi.org/project/pycontractions/

Rao, K. Y., Murthy, G. S., & Adinarayana, S. (2017). Product Recommendation System from Users Reviews using Sentiment Analysis. *International Journal of Computer Applications, 169*(1), 30-37.

Reyes, A., & Rosso, P. (2012). Making objective decisions from subjective data: Detecting irony in customer reviews. *Decision Support Systems, 53*(4), 754-760.

Scikit-learn. (2018, October 28). *LogisticRegression*. Retrieved October 2018, from Scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRegression

Scikit-learn. (2018, November 15). *Machine Learning in Python*. Retrieved November 2018, from Scikit-learn: https://scikit-learn.org/stable/modules/classes.html

Scikit-learn. (2018, November 20). *Multiclass and multilabel algorithms*. Retrieved November 2018, from Scikit-learn: https://scikit-learn.org/stable/modules/multiclass.html

Scikit-learn. (2018, November 3). *MultiLabelBinarizer*. Retrieved November 2018, from scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MultiLabelBinarizer.html

Scikit-multilearn. (2018, November 18). *Scikit-multilearn*. Retrieved 18 2018, from Scikit-multilearn API Reference: http://scikit.ml/api/skmultilearn.html

Szymański, P., & Kajdanowicz, T. (2017). A scikit-based Python environment for performing multi-label classification. *arXiv preprint* , arXiv:1702.01460.

Tan, P.-N. (2006). *Introduction to data mining.* Boston: Pearson Addison Wesley.

Thakkar, H., & Patel, D. (2015). Approaches for Sentiment Analysis on Twitter: A State-of-Art study. *arXiv:1512.01043 [cs.SI]*.

Tkachenko, M., & Lauw, H. W. (2014). Generative modeling of entity comparisons in text. *Proceedings of the 23rd acm international conference on conference on information and knowledge management* (pp. 859-868). ACM.

Tsoumakas, G., & Katakis, I. (2007). Multi-Label Classification: An Overview. *International Journal of Data Warehousing and Mining, 3*(3), 1-10,12-13.

Tsoumakas, G., Katakis, I., & Vlahavas, I. (2009). Mining multi-label data. In L. R. Oded Maimon, *Data Mining and Knowledge Discovery Handbook* (pp. 667-685). Boston, MA: Springer.

Turney, P. D. (2002). Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. *Proceedings of the 40th annual meeting on association for computational linguistics* (pp. 417-424). Philadelphia, Pennsylvania: Association for Computational Linguistics.

Turney, P. D., & Littman, M. L. (2003). Measuring Praise and Criticism: Inference of Semantic Orientation from Association. *ACM Transactions on Information Systems (TOIS), 21*(4), 315-346.

Unnisa, M., Ameen, A., & Raziuddin, S. (2016). Opinion Mining on Twitter Data using Unsupervised Learning Technique. *International Journal of Computer Applications, 148*(12), 12-19.

Vaghela, V. B., & Jadav, B. M. (2016). Analysis of Various Sentiment Classification Techniques. *Analysis, 140*(3), 22-27.

Wang, W., Feng, Y., & Dai, W. (2018). Topic analysis of online reviews for two competitive products using latent Dirichlet allocation. *Electronic Commerce Research and Applications, 29*, 142-156.

Wang, W., Xin, G., Wang, B., Huang, J., & Liu, Y. (2017). Sentiment information Extraction of comparative sentences based on CRF model. *Computer Science & Information Systems, 14*(3), 823-837.

Varathan, K. D., Giachanou, A., & Crestani, F. (2017). Comparative Opinion Mining: A Review. *Journal of the Association for Information Science and Technology, 68*(4), 811-829.

Varghese, R., & Jayasree, M. (2013). A survey on sentiment analysis and opinion mining. *International Journal of Research in Engineering and Technology, 2*(11), 312-317.

Wei, Z., Zhang, H., Zhang, Z., Li, W., & Miao, D. (2011). A Naive Bayesian Multi-label Classification Algorithm With Application to Visualize Text Search Results. *International Journal of Advanced Intelligence, 3*(2), 173-188.

Vens, C., Struyf, J., Schietgat, L., Džeroski, S., & Blockeel, H. (2008). Decision trees for hierarchical multi-label classification. *Machine Learning, 73*(2), 185-214.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.

Vohra, S. M., & Teraiya, J. B. (2013). A comparative study of sentiment analysis techniques. *Journal JIKRCE, 2*(2), 313-317.

Xu, K., Liao, S. S., Li, J., & Song, Y. (2011). Mining comparative opinions from customer reviews for Competitive Intelligence. *Decision support systems, 50*(4), 743-754.

Ye, Q., Zhang, Z., & Law, R. (2009). Sentiment classification of online reviews to travel destinations by supervised machine learning approaches. *Expert systems with applications, 36*(3), 6527-6535.

Zhang, M.-L., & Zhou, Z.-H. (2007). M L-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition, 40*(7), 2038-2048.

Zhang, M.-L., & Zhou, Z.-H. (2014). A Review on Multi-Label Learning Algorithms. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, 26*(8), 1819-1837.

Zhu, J., Wang, H., Zhu, M., Tsou, B. K., & Ma, M. (2011). Aspect-Based Opinion Polling from Customer Reviews. *EEE Transactions on Affective Computing, 2*(1), 37-49.

Zhuang, L., Jing, F., & Zhu, X.-Y. (2006). Movie Review Mining and Summarization. *Proceedings of the 15th ACM international conference on Information and knowledge management* (pp. 43-50). ACM.