

Improving Sentiment Analysis by Detecting Light Verb Construction

By

Liew Er Wei



**FACULTY OF COMPUTING AND
INFORMATION TECHNOLOGY**

**TUNKU ABDUL RAHMAN UNIVERSITY COLLEGE
KUALA LUMPUR**

**ACADEMIC YEAR
2019/20**

Improving Sentiment analysis by detecting Light Verb Construction

By

Liew Er Wei

Supervisor: Ts. Jessie Teoh Poh Lin

A project report submitted to the
Faculty of Computing and Information Technology
In partial fulfillment of the requirement for the
Bachelor of Computer Science (Honours)

Department of Information and Communication Technology
Faculty of Computing and Information Technology
Tunku Abdul Rahman University College
Kuala Lumpur

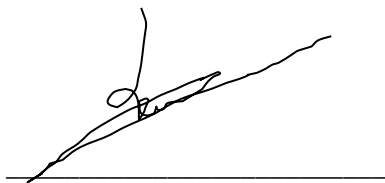
2019/20

Copyright by Tunku Abdul Rahman University College.

All rights reserved. No part of this project documentation may be reproduced, stored in retrieval system, or transmitted in any form or by any means without prior permission of Tunku Abdul Rahman University College.

1. Declaration

The project submitted herewith is a result of my own efforts in totality and in every aspect of the project works. All information that has been obtained from other sources had been fully acknowledged. I understand that any plagiarism, cheating or collusion or any sorts constitutes a breach of TAR University College rules and regulations and would be subjected to disciplinary actions.

A handwritten signature in black ink, appearing to read 'Liew Er Wei', is written over a horizontal line.

Liew Er Wei

Bachelor of Computer Science (Honours) in Software Engineering

ID: 19WMR09766

2. Abstract

Sentiment analysis is a computational study of people's views, attitudes, and emotions on entities. This project aim to improve the accuracy of sentiment analysis by using Multiword Expressions to detect the light verb construction in the sentence. The Multiword Expressions are made up of at least two words and which can be syntactically or semantically idiosyncratic in nature. The algorithm of sentiment analysis will be help to performing function to classify the sentiment of comments into six categories, which are anger, sad, happiness, fear and disgust. The sentiment analysis can help to people to make decision based on the result of sentiment analysis. The selected algorithm is Linear Support Vector Machine (SVM), Naïve Bayes, and Logistic Regression algorithm to solve the sentiment analysis to classify the sentences. Before that, this paper also will be compare the performance of each algorithm. However, in order to improve the accuracy of sentiment analysis, it is necessary to develop Multiword Expressions. Multiword Expressions constructed by light verbs in sentences are detected by using the spaCy library to perform rule based matching approach. As the result, the project prove that the light verb construction extraction can improve the accuracy of the sentiment analysis in three model. The project will be using the cross-industry standard process for data mining (CRISP-DM) methodology to process. The CRISP-DM provide a structure approach and an overview of the life cycle to help planning a data mining.

3. Acknowledgement

I would like to take this opportunity to express my gratitude and gratitude to my supervisor, Ts. Jessie Teoh Poh Lin has given great support, feedback and guidance to me throughout the project implementation process, to make sure the project is doing the right way. In addition, I also would like to thank my team members for continuously motivating and encouraging me, during I encounter difficulties, they can ensure the project complete successfully.

4. Table of Contents

1. Declaration	ii
2. Abstract	iv
3. Acknowledgement	v
4. Table of Contents	v
1. Introduction	8
1.1 Problem Statement	9
1.2 Project Objectives	10
1.3 Advantages and Contribution	11
1.4 Project Plan	12
1.4.1 System Structure	12
1.4.2 Modules	12
1.4.3 Project Schedule	13
1.4.4 Methodology	14
1.4.5 Development and Operation Environment	15
1.5 Project Team and Organization	16
1.6 Chapter Summary and Evaluation	17
2. Project Background & Literature Review	19
2.1 Project Background	19
2.1.1 Target Market	19
2.2 Literature Review	20
2.2.1 Sentiment Analysis	20
2.2.2 Sentiment Analysis Algorithm	23
2.2.3 Machine Learning	25
2.2.4 Supervised Machine Learning	26
2.2.5 Support Vector Machine (SVM)	27
2.2.6 Naive Bayes	29
2.2.7 Logistics Regression	30
2.2.8 Natural Language Processing (NLP)	31
2.2.9 Text Processing	32
2.2.10 Six Basic Emotion Model	35
2.2.11 Multiword Expression	37
2.2.12 Light Verb Construction	42
2.2.13 Cross-Industry Standard Process for Data Mining	47
2.3 Chapter Summary and Evaluation	49
3. Methodology and Requirements Analysis	51
3.1 Business Understanding	52
3.1.1 Determine Data Mining Goals	52
3.1.2 Produce Project Plan	52
3.2 Data Understanding	53
3.2.1 Collect Initial Data	53
3.2.2 Describe Data	54
3.2.3 Explore Data	56
3.2.4 Verify Data Quality	59
3.3 Data Preparation	60
3.3.1 Select Data	60
3.3.2 Clean Data	61
3.3.3 Format Data	68
3.4 Modeling	69

3.4.1	Select Modeling Technique.....	69
3.4.2	Generate Test design	69
3.4.3	Build Model	70
3.4.4	Assess Model	72
3.5	Evaluation	87
3.5.1	Evaluate Result.....	87
3.5.2	Review Process	87
3.6	Deployment.....	88
3.6.1	Plan Deployment.....	88
3.7	Functional Requirement.....	89
3.8	Non-Functional Requirement	89
3.9	Chapter Summary and Evaluation	90
4.	System Design	92
4.1	User Interface Diagram.....	92
4.2	System Architectural Design	94
4.3	Activity Diagram	96
4.4	Chapter Summary and Evaluation	98
5.	Implementation and Testing	100
5.1	Implementation	100
5.1.1	Pickle File	101
5.1.2	Backend	104
5.1.3	Text Pre-processing	108
5.1.4	Multiword Expression implement.....	109
5.1.5	Classification Model	111
5.1.6	Web Page	113
5.1.7	Bar chart	116
5.1.8	Upload to CentOS server.....	117
5.2	Testing	118
5.2.1	Unit Testing.....	118
5.2.2	Integration Testing	118
5.2.3	System testing	118
5.2.4	Test Case	119
5.3	Testing	121
5.4	Chapter Summary and Evaluation	123
6.	MWEToolKit and.....	125
6.1	Multiword Expression Toolkit (MWEToolKit).....	125
6.1.5	Component of MWEToolKit	127
6.2	Ngram Statistic Packages.....	128
6.3	Chapter Summary and Evaluation	131
7.	Discussions and Conclusion.....	133
7.1	Summary	133
7.2	Achievements.....	133
7.3	Contributions	134
7.4	Limitations and Future Improvements.....	134
7.5	Issues and Solutions.....	135
8.	References	136
9.	Appendices	140

Chapter 1

Introduction

1. Introduction

With the rapid popularity of social networks, and forums, the role of sentiment analysis has grown significantly (Shah, 2018). Today, almost every webpage has a section for users to leave comments about products or services and share them with friends on Facebook, Twitter and other, which was impossible a few years ago (Shah, 2018). The emotion will be affect our quality of life and the way we interact with others. They determine our thinking, the action we take, our subjective understanding of the world, and our behavioural responses. Before that, text also contain large amount of opinion and emotion of people about the perception of product and services. The sentiment analysis is a computational study of people's views, attitudes, and emotions on entities. Moreover, sentiments analysis also used in different field, such as economic, or political decisions, and assistive technology (Balan et al., 2019).

Sentiment analysis is the task of performing natural language processing (NLP) that to identify the sentiments they express and classify the people emotion into six basic emotion model based on their comment or text document. The six basic emotion model is a popular emotion recognition model on computers. It distinguishes six basic emotions: anger, disgust, fear, happiness, sadness, and surprise. However, the existence of Multiword Expressions (MWEs) will become a key problem and a huge challenge in the process of sentiment analysis (Sag et al., 2002). This is because that the Multiword Expressions (MWEs) is more complicated, and it contains different semantics to express certain emotions. Not only that, research found that the structure of Multiword expressions (MWEs) are related to the intensity of emotion denoted (Fotopoulou, Aggeliki & Giouli, Voula, 2018). Therefore, in order to ensure sentiment analysis has high performance and accuracy, the Multiword Expressions (MWEs) detection must be apply to detect the light verb in the text sentences.

1.1 Problem Statement

The sentiment analysis task is considered to be a sentiment classification problem. Based on the research found that in recent years, many applications and enhancements of sentiment analysis algorithms have been proposed. Sentiment analysis is the task of performing natural language processing (NLP) to determine people's attitudes and opinions on different topics, products, services, events and their attributes. By using the sentiment analysis, it can bring many benefits to the company, so that the company can make any improvements in the future to increase the acceptance of customers. However, the existence of Multiword Expressions (MWEs) will become a key problem and a huge challenge in the process of sentiment analysis. This is because Multiword Expressions (MWEs) is more complicated, and it contains different semantics to express certain emotions. In this process, the accuracy of the sentiment analysis results may be affected. Prior to this, the user's comments usually included the user's behavior and intentions to the post, and in user's comments also contain many irrelevant or less important word for sentiment analysis such as username, punctuation, symbol, specific character and other. Not only that, it was difficult to understand the user's needs and identify user's emotion expressed by manually reading the comments one by one. At the same time, it was very slow speed, inefficient and low accuracy. Therefore, in order to solve this problem, it necessary to implement sentiment analysis with Multiword Expressions (MWEs) detection to detect the light verbs to make sure the sentiment analysis can be perform well to provide high accuracy, faster response speed and more effective and efficient result.

.

1.2 Project Objectives

The main objective of the project is develop a sentiment analysis system with detect the light verb construction, in order to improve the accuracy. The sentiment analysis system will be develop as a Web based system to allow users can run the system in browser. Supervised machine learning is a popular approach and most frequently to use for text classification purpose in sentiment analysis, such as Support Vector Machine (SVM), Naïve Bayes, and Logistic Regression algorithm. The sentiment analysis system will using the supervised machine learning algorithm to perform the sentiment analysis function to classify the emotion based on the text sentence into six basic emotions. The sentiment analysis system also consist the Multiword Expressions (MWEs) detection, which is light verb construction detection. The light verb construction detection can detect the light verb in text sentence and replaced by a single word with the same meaning to help the sentiment analysis system to improve the accuracy during the process of predict the emotions of the test sentences. In order to achieve Multiword Expression (MWEs) detection, natural language processing (NLP) require to apply. Natural language processing is a field of research and application. It explores how to use a computer to learn natural language text to do useful things. In encoding process, is need to use a natural language processing library. This is because that it help to simplify text processing, allowing them to focus on building machine learning algorithm, such as lemmatization, tagging, parsing, semantic reasoning and tokenization (B.Chrystal and Joseph, 2015). Based on the research, there are lack of research about the light verb construction related to the sentiment analysis, therefore, propose the compare the performance of extract light verb construction and without extract light verb construction, to determine the detect the light verb construction can help to improve the sentiment analysis. Moreover, some researchers are not consider about the light verb construction consist of adjective and adverb ().

In short, the objective of this project:

- To identify the sentiment from the comments
- To detect the light verb construction in sentences
- To improve the accuracy of the sentiment analysis
- To compare the performance of sentiment analysis algorithms
- To develop the web-based system for sentiment analysis

1.3 Advantages and Contribution

Most companies are always overwhelmed by the need for Sentiment Analysis systems. This is because the system can help them analyze and detect the sentiment in the comment text sentences obtained from the social media platform such as Facebook, Twitter, and other platforms. By using sentiment analysis, the company can understand the customer's satisfaction and emotion with its products or services. Examples include anger, disgust, fear, happiness, sadness, and surprise. With this information, the company can make necessary changes or improvements to its products or services. As a result, companies can increase their customer satisfaction and acceptance to improve business performance. Moreover, the sentiment analysis can recognize and manage Multiword Expressions (MWEs) in notes. For example, light verb construction, it may contain many emotions and present different emotions. Therefore, the detection of Multiword Expressions (MWEs) can help the system improve the accuracy of sentiment analysis.

In addition, this project will develop as HTML website platform, it allow user to use it in the browser. Users can upload the file to system in order to perform sentiment analysis then the user can be get the sentiment result from the system. This can help the user or the company get the result in the easy way and they can make the decision making based on the result. This also bring the benefit to them to save the time and the website platform can provide users with a familiar service to guide the user and make user easy to use the system.

1.4 Project Plan

1.4.1 System Structure

In this project structure chart has 2 main modules as shown in figure 1.1, which are sentiment analysis algorithms (SA) and Multiword Expression (MWE). In sentiment analysis algorithms (SA), it consists of support vector machine (SVM), Naïve Bayes and Logistic Regression algorithms. And in Multiword Expression (MWE), it consist the light verb construction.

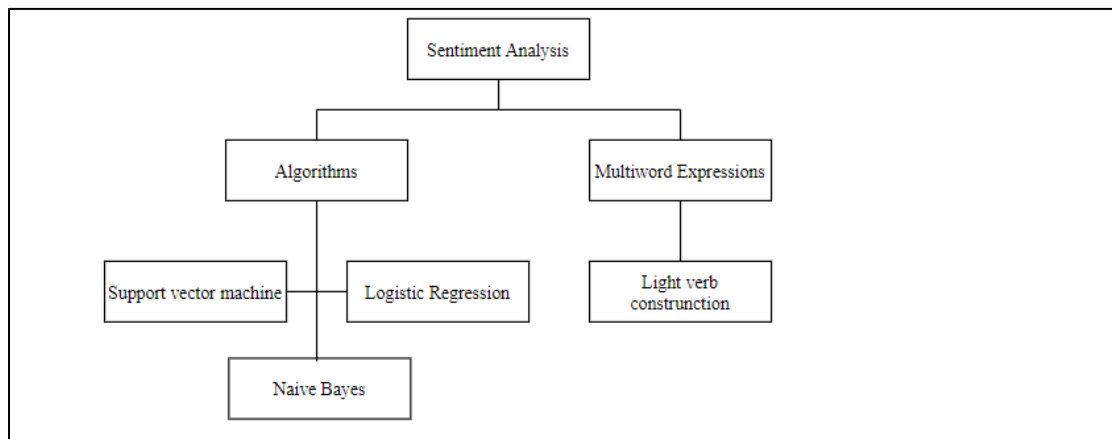


Figure 1.1: system structure chart

1.4.2 Modules

Sentiment Analysis Algorithm Modules

1. Support Vector Machine (SVM): To classify the sentiment by using the Support Vector Machine (SVM) algorithm.
2. Naïve Bayes: To classify the sentiment by using the Naïve Bayes algorithm.
3. Logistic Regression: To classify the sentiment by using the Logistic Regression algorithm.

Multiword Expression Modules

1. Light verb construction: Detect light verb construction, and replacement on text with same meaning single word.

1.4.3 Project Schedule

The schedule of this project as shown in figure 1.2 and table 1.1

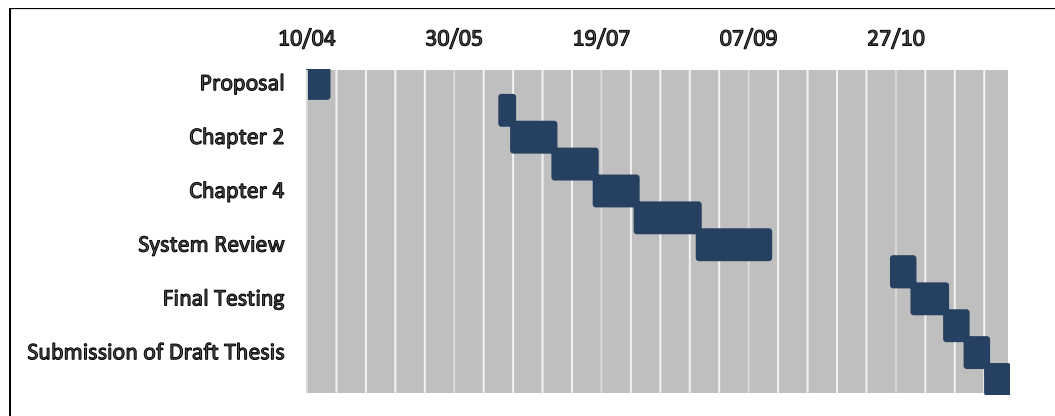


Figure 1.2: Gantt chart of the sentiment analysis project

Milestone	Milestone goal	Deadline
Proposal	Approach supervisor Finalize project proposal	17/04/2020
Chapter 1	Introduction Project specification Research Area	19/06/2020
Chapter 2	Literature review Project/System background Preliminary research or feasibility study	03/07/2020
Chapter 3	Research Approaches Research or development model Requirements Analysis	17/07/2020
Chapter 4	System/software Design Specifications Algorithms	31/07/2020
Submission Project I	Submission of Project I	21/08/2020
System Preview	To preview the prototype to the supervisor	14/9/2020
Test Plan Preparation	Preparation of test plan	02/11/2020
Final Testing	Testing with Supervisor and Moderator	13/11/2020
Make Good testing	Make Good Testing for student who did not fulfil the requirement to pass	20/11/2020
Submission of Draft Thesis	Submission of Draft FYP report	27/11/2020
Submission of Final Thesis	Submission of Final FYP report	4/12/2020

Table 1.1: milestones of the sentiment analysis project

1.4.4 Methodology

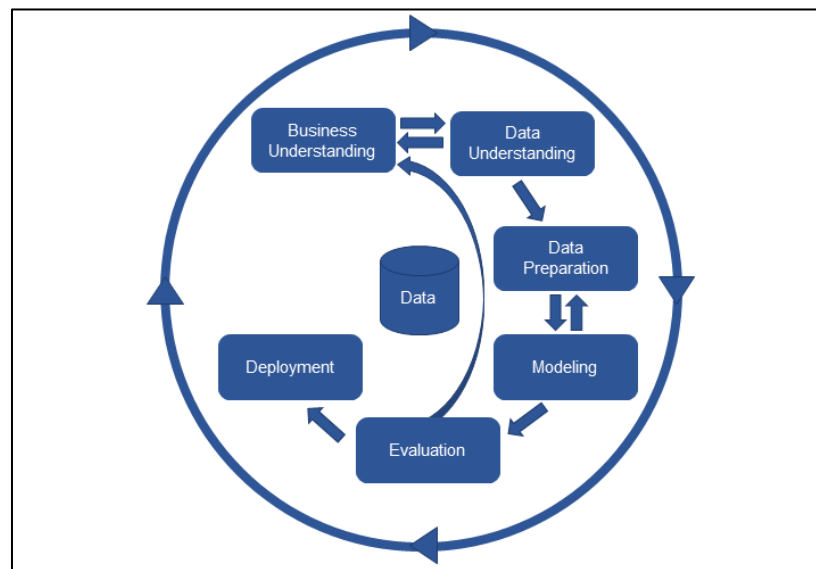


Figure 1.3: phase of life cycle of the CRISP-DM process model

The methodology use in this project is cross-industry standard process for data mining (CRISP-DM). The data mining is a creative process which requires a several skills and knowledge. The cross-industry standard process for data mining methodology can be provides a structured approach and an overview of the life cycle in order to planning a data mining in this project. The life cycle of the CRISP-DM can refer the figure 1.3. The life cycle of the CRISP-DM is broken down in six phases, which is business understanding, data understanding, data preparation, modelling, evaluation and deployment.

The initial phases of CRISP-DM aim to focus the understanding the project objective and find factor that influence the outcome of the project. CRISP-DM start to data collection and process activities in order to get the data in data understanding phases such as to apply the twitter developer account to gather the data. And cover the data into dataset in data preparation phase. The modelling phase, will be select and build the model technique, this project is use the sentiment analysis algorithms, such as Support Vector Machine (SVM), Naïve Bayes and Logistic Regression algorithms to classify the sentiment analysis. In evaluation phase, will evaluate the model, and review the steps executed to construct the model, to be certain it properly achieves the objectives. And in last phase, will generating a report and present to the supervisor and moderator.

One of the basic decisions was to follow the CRISP-DM methodology as much as possible. By using the CRISP-DM process model for planning the case study, for communication within the project team, for communication with the supervisor, and for documenting results and experiences. Before that, the CRISP-DM process model helps to learn about and to understand prospects. Therefore, to get an understandable, and actionable profile as a further requirement.

1.4.5 Development and Operation Environment

Development Environment

This project will be developed in own development environment. The components in this project such as framework, software, hardware, operating system and programming languages to ensure the sentiment analysis system can be used.

Framework	: Flask
Software	: csv file, Spyder, Jupyter notebook
Hardware:	: Laptop / desktop
Operating system	: Microsoft Window 10
Programming Languages	: HTML, CSS, Python

Operational Environment

The sentiment analysis system allow the user to use laptop or desktop computer to implement and use. The system will be developed as HTML platform, therefore the hardware device must use the web browser to run the system such as Google Chrome, Mozilla Firefox, and Microsoft Internet Explorer. User allow to upload the file with consist comments from the social media platform. Then, the system will be display the result of each comment on the screen and the user can also get the output file from the system.

1.5 Project Team and Organization

The Project will be handled by four project team members as shown in table 1.2, which are Liew Er Wei, Phang Jia Luo, Low Hong Ming, and Fung Wai Lun. Besides, this project also will be supervised by Ms. Jessie Teoh Poh Lin to guide the project team members to make sure this project can be complete successful.

Project team members	
1.	Liew Er Wei
2.	Phang Jia Luo
3.	Low Hong Ming
4.	Fung Wai Lun

Table 1.2: Project team of sentiment analysis project

1.6 Chapter Summary and Evaluation

In this chapter, sentiment analysis will be plan to develop, in order to help the users to identify the people sentiment in the efficiency way, to save their time. Based on the sentiment analysis system, users are no necessary to perform sentiment analysis manually, it is take more time to work and easy to make a mistake to identify wrong sentiment, to affect the accuracy. In order to make sure the sentiment analysis have high accurate, the Multiword Expressions (MWEs) detection must be implement which is light verb construction detection such as 'do the work', 'make a decision' and 'take a sleep'. Based on the research, Multiword Expressions (MWEs) have relationship related to people emotions, therefore it will be huge challenge face in sentiment analysis. The light verb construction detection will be detect the light verb construction in the sentence and replace with same meaning single word. For example, 'make a decision' will replace to 'decide'. It can make sentence shorter and remove some no importance words.

Chapter 2

Project Background & Literature Review

2. Project Background & Literature Review

2.1 Project Background

This project is focusing the multiword expression detection, which is light verb construction detection and it also focusing the sentiment analysis. The light verb construction detection is using the natural language processing (NLP) to detect the light verb constructions in the sentence, and it will replace the light verb construction to a single word with same meaning. This is because that, the light verb can usually be explained with a simple verb. Before that, the dataset will be collect from the Twitter platform and this dataset will be processed the light verb construction detection then will process further analysis, which is sentiment analysis. The sentiment analysis will be using the support vector machine (SVM), Naïve Bayes, Logistic Regression algorithm to classify the sentiment of comment in the dataset.

2.1.1 Target Market

The target market of the system can be any company that they want to analyses customers' sentiment or perceptions of their products or services, so they can make improvement their business performance. According on current research, found that many companies are lack attention to customer requirement and do not perform sentiment analysis on their product comments or reviews on social media platforms. Therefore, sentiment analysis system can help companies to perform tasks, to save their time taken and reduce the workload of reading each comment on the social media platform. Not only that, based on the result generated by sentiment analysis system, the companies can understand their customers' preferences and know which areas or services they need to improve. In this way, the company can fully meet the customer requirement and maintain customer loyalty.

2.2 Literature Review

2.2.1 Sentiment Analysis

The emotion will be affect our quality of life and the way we interact with others. They determine our thinking, the action we take, our subjective understanding of the world, and our behavioural responses. Sentiment analysis is the task of performing natural language processing (NLP) to determine people's attitudes and opinions on different topics, products, services, events and their attributes. Sentiment analysis can be seen as a process, including the following steps: define the purpose and set goals, then perform data retrieval, pre-processing, feature extraction, text classification based on sentiment, and finally evaluate the results (Buzic, Dalibor, 2019).

Although the sentiment analysis can be performed in image recognition, speech recognition, or in combination of image and text, only the textual content will be used for sentiment analysis in this project.

In the past, people have been relying on these suggestions, but most of them are limited to the opinions of friends and family. Now, people can read a large number of opinions or comments of strangers on social media. With the rapid popularity of social networks, and forums, the role of sentiment analysis has grown significantly (Shah, 2018). Today, almost every webpage has a section for users to leave comments about products or services and share them with friends on Facebook, Twitter and other, which was impossible a few years ago (Shah, 2018).

Twitter is one of the most popular social media platforms. In the past ten years, it has been growing steadily until now it has been use by 330 million active users and has become a meeting point for various groups of people: students, professionals, celebrities, companies, politicians. This popularity of Twitter results in the large amount of information being pass through the service, covering the opinions of brands, products, politicians, and social activities related to the well-being of people from a wide range of subjects (Shah, 2018). In this case, Twitter becomes a powerful tool for predictions. For example, researchers try to predict the ticket sales level of movies based on Twitter information. The team successfully predicts the revenue of the

opening weekend with 97.3% accuracy, which is higher than the prediction rate of the Hollywood stock exchange, a known movie prediction tool (Shah, 2018).

In research literature, it is possible to see many different names, such as sentiment analysis, opinion mining, opinion extraction, sentiment mining, subjectivity analysis, emotion analysis, review mining (Shah, 2018). However, they all have similar purposes and belong to the subject of sentiment analysis or opinion mining. The purpose of sentiment analysis is to improve user experience and quality of life, which is why various emotion models have been proposed over the years and effective mathematical models have been applied to extract, classify and analyse emotions (Balan et al., 2019). Sentiment analysis is a computational study of text evaluation expression. Information systems have brought many advantages on the social media platforms. These information systems can extract information about people's event, opinions, and emotions about events, individuals, organizations, products, services, or topics (Alnawas and Arıcı, 2018).

The task of sentiment analysis is to find people's views on specific entities in the text, which is a technique to classify people's views. An important concept in sentiment analysis is to obtain semantic orientation of the words, which refers to the sentiment polarity and sentiment intensity of words, phrases or texts (Alnawas and Arıcı, 2018). For example, 'pretty' is labelled as positive, while the "ugly" is labelled as negative. The goal of sentiment analysis is usually to find the semantic direction of the text (Alnawas and Arıcı, 2018).

Mining a large number of opinions can provide information for understanding collective human behaviour (Shah, 2018). More and more evidence points out that by analysing the sentiment of social media content, it may be possible to predict the size of the market, marketing activities, and marketing results (Shah, 2018). Therefore, many companies are aware of this, and sentiment analysis can help them understand how a product's reputation changes over time, such as the product's weakness.

The snippets of reviews are a gold mine for companies and individuals who want to monitor reputation and get timely feedback on their products and behaviours (Alnawas and Arıcı, 2018). Sentiment analysis enables these organizations to monitor different

social media sites in real-time and act accordingly (Alnawas and Arıcı, 2018). Marketing managers, public relations companies, campaign managers, politicians, and even stock investors and online shoppers are the direct beneficiaries of sentiment analysis technology. From a less commercial perspective, health and education services, police, the judiciary, and any counterterrorism organization can also be added to the list of "beneficiaries of sentiment analysis technology" (Alnawas and Arıcı, 2018).

These are some examples in a number of areas of sentiment analysis. Between 73% and 87% of people claim that reviews of restaurants, hotels, and travel agencies have a significant impact on purchases (Balan et al., 2019). For example, buyers are willing to pay more for five-star products or services than four-star products or services cost. Sentiment analysis is also famous in the medical field. For example, a personalized recommendation system for analysing user reviews and physician attributes to understand customer preferences for drug pricing. Forecasting stock price movements is very popular in the research community. Sentiment analysis of Twitter messages has shown that the polarity of the news may be an indicator of stock price changes a few days in advance (Balan et al., 2019).

These are just a few examples in a number of areas of sentiment analysis. Reviews of social events, marketing campaigns, and company strategies are also often used for sentiment analysis.

2.2.2 Sentiment Analysis Algorithm

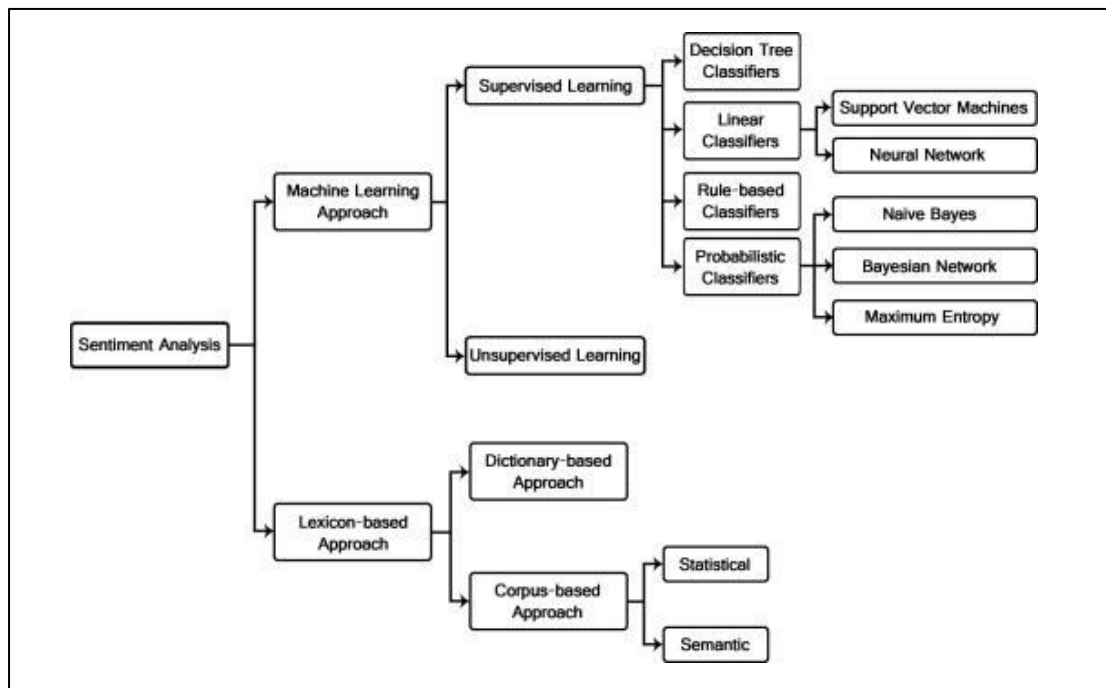


Figure 2.1: sentiment analysis algorithms

Sentiment analysis algorithm can be divided into 3 main approaches, which are machine learning approach, lexicon based approach and hybrid approach (refer figure 2.1).

The Machine Learning approach is implement the famous machine learning algorithms and use the linguistic feature to classify the class labels. The machine learning approach have two different types, which are supervised and unsupervised machine learning. The supervised machine learning is depend on the existence of labelled training dataset to predict the expected outcome. The unsupervised machine leaning are use when it is difficult to find labelled training dataset. In the machine learning approach, supervised machine learning is the most frequently used to classifying emotion in sentiment analysis and supervised machine learning will be used for sentiment analysis in this project (Tripathy and Rath, 2017).

The lexicon-based approach are relies on a sentiment lexicon, a collection of know and precompiled sentiment term. Lexicon-based approach also can know opinion lexicon. Opinion words are apply in many sentiment analysis classification task. The opinion

can be positive or either negative. Positive opinion word is used to express desired states, while negative opinion word are opposite the positive opinion word; it is used to express undesired states. There are three approaches to collect or compile the opinion word list. First approach is manual approach, this approach normally is not use alone and it is very time consuming. Then it is combine with other two automated approaches from lexicon-based approach (Kolchyna et al., 2015). The lexicon-based approach have two types, which are dictionary-based approach and corpus-based approach. The dictionary-based approach are use a small set of manually collected words of known sentiment orientation that is then expanded using a synonyms and antonyms dictionary to create a word list such WordNet library model. The Corpus-based approach will be begin with a list of words of known orientation and can be expanded with the syntactic patterns, which are use the semantic or statistical method to find and know sentiment polarity model (Kolchyna et al., 2015).

The hybrid approach is combines two approaches to perform the sentiment analysis, and common with sentiment lexicons playing a key role in the majority of methods. Based on the research, this approach is not yet frequent used for sentiment analysis because its computational complexity is higher (Kolchyna et al., 2015).

2.2.3 Machine Learning

The machine learning has become the development center of mainstream information technology. With more and more data become a sufficient reason to believe that intelligent data analysis will become more prevalent necessary element of technological progress.

Machine learning can be used for perform the sentiment analysis. Machine learning is a field of computer science, which is the computer that predicts the next task to perform by analysing the data provided to it (Tripathy and Rath, 2017). As computing technologies are improved from time to time, machine learning system nowadays unlike the system in the past and it was designed to automatically apply complex mathematical calculations on a large data set in an extremely short time while still able to maintain high accuracy(Tripathy and Rath, 2017).

In general, machine learning approach suitable for this problem are usually supervised classifications, especially text classification techniques for opinion mining.

In machine learning approach, most use a set of features containing n-grams to train the classifier. Based on the research, researchers have shown that machine learning classifiers outperform lexicon-based approach. However, this superiority is usually limited to a single domain (Ghiassi and Lee, 2018).

The machine learning can be categorized into supervised, semi-supervised and unsupervised machine learning approach. In this project, the supervised machine learning approach used for text classification in sentiment analysis.

2.2.4 Supervised Machine Learning

The supervised machine learning approach are popular in classification problems such as, whether a text contains number or character; classifying weather conditions; a feedback is positive or negative and classifying gender of human beings; sentiment classification and so on (Choudhari, P. & S, V.D. 2017).

Supervised machine learning approaches are widely used for classification purposes. In this technique, model or learner is first trained with some sample data which has been assigned to the category and then test the model by using some sample data as input to the model for doing classification based on the prior training given to it. (Tripathy and Rath, 2017). The performance of the model is measured according to the accuracy, precision, and recall of the categorization.

Supervised machine learning has many type algorithms can be used for sentiment analysis such as decision tree, random forest, neural network, only the support vector machine (SVM), Naive Bayes and Logistic Regression the will discuss in this project.

2.2.5 Support Vector Machine (SVM)

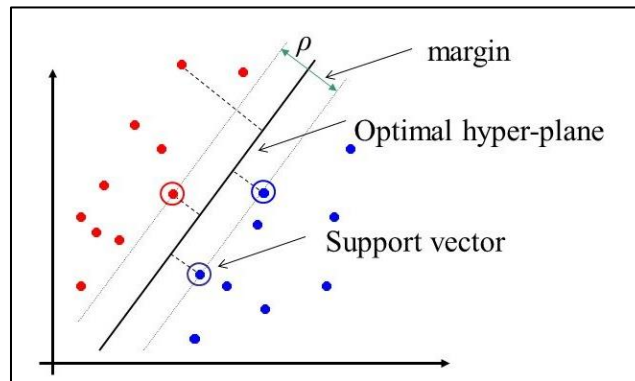


Figure 2.2: hyperplane of Support Vector Machine (SVM)

The main principle of support vector machine (SVM) is to determine the linear delimiter in the search space to best separate the different classes. SVM is a non-probabilistic linear classifier, which can construct a nonlinear decision surface in the original feature space by nonlinearly mapping data instances to an internal product space, in which hyperplane linear separation can be used Categories to classify data (Medhat, Hassan and Korashy, 2014). SVM classification due to the sparsity of text, in which few features are irrelevant, but they are often related to each other and are usually organized into linearly separable categories (Kolchyna et al., 2015)

SVM can classify text by drawing a hyperplane in a scatterplot between positive and negative class labels as shown in figure 2.2. The representative of the document closest to the hyperplane is called a support vector. Here, the goal of SVM is to find an optimal separating linear hyperplane that maximizes the margin between data points marked by two classes (Chang and Huo, 2018). In research, SVM has been proven very effective in traditional text classification, which is usually superior to the Naive Bayes approach (Ghiassi and Lee, 2018).

Based on various research, most cases the classification report point out that the SVMs algorithms provide the best performance in terms of accuracy, f1 score, precision and other compared with other supervised machine learning such as naïve bays, decision tree, and random forest.

SVMs are used in many applications, where these applications classify reviews based on their quality. There two common multiclass SVM-based approaches, One-versus-All SVM and One-versus-One SVM. The One-versus-All SVM also known as OVA classifier and One-versus-One SVM also can know as the OVO classifier. These approaches can be used for classify reviews. Based on literature research, researchers have used these methods to assess the quality of information in product reviews and treat it as a classification problem. They are worked on the digital camera and MP3 review. "Their results show that their methods can accurately classify reviews based on the quality of the reviews and show that it is much better than the state-of-the-art methods (B.Chrystal and Joseph, 2015).

2.2.6 Naive Bayes

Naive Bayes algorithm are probabilistic classifiers that apply Bayes' theorem with naive independence presumption amongst the features (Arulmurugan, Sabarmathi and Anandakumar, 2017). Naive Bayes algorithm also is the simplest and most commonly used for classification in sentiment analysis. The Naive Bayes algorithm computes the posterior probability of a class, based on the distribution of the words in the document.

The Naïve Bayes algorithm is a simple technique employed for classifying the class labels. It is assign the problem instances with class labels, like positive or negative to represent as vectors of feature values, where the class tags are drawn out of some finite set (Arulmurugan, Sabarmathi and Anandakumar, 2017).

The Naive Bayes algorithm is called Naive because it considers each feature to be the same under the assumption of word independence. For example, the words have no dependency or connected with other words in the document being considered for classification purposes (Choudhari, P. & S, V.D. 2017). Therefore, this means that all attributes or words are independent, and can explain, as one word does not affect the other in deciding whether the tweet or review is positive, negative or other class labels (Choudhari, P. & S, V.D. 2017).

Even many research shows that the result of the Support Vector Machine (SVM) algorithm is better than the Multinomial Naïve Bayes algorithm. However, the Multinomial Naïve Bayes algorithm are faster time taken in training and predict process, it is the most suitable use when user need the result quickly. Before that, Naïve Bayes also can provide good performance in the analysis the short document and it is better in snippet.

2.2.7 Logistics Regression

Logistic Regression is a discriminative classification algorithm that can generate the probability values of data belonging to different class labels. It is form of regression that allows the prediction of outcome variables through a combination of continuous and discrete predictors. Compared with Naive Bayes algorithm, which has strong conditional independence between variables, logistic regression algorithm has better results. The logistic function use in logistic regression algorithm also known as sigmoid function, it maps predicted values to probability values (Prabhat and Khullar, 2017). The logistic regression was seen as similar to the linear regression, however, the linear regression was used for the regression function, while the logistic regression was used for the classification function.

There are three types of logistic regression algorithms, which are binary, multi and ordinal class logistic regression algorithms, it perform is depend on the type of target class. In the case of this project, multinomial logistic regression may most suitable use for classifying and predict the class label.

The multinomial logistic regression algorithm, also known as the softmax regression due to the use of hypothesis functions, which can be used to solve several problems including text classification, which can use in the sentiment analysis. Multinomial logistic regression algorithms can generalize logistic regression to classification problems, where the outcome can take more than two targets (Ramadhan, Astri Novianty and Casi Setianingsih, 2017).

The literature review shows that the multinomial logistic regression algorithms require more time to be trained compared with the multinomial Naive Bayes algorithm; this is because it uses an iterative algorithm to estimate the parameters of the model (Ramadhan, Astri Novianty and Casi Setianingsih, 2017).

2.2.8 Natural Language Processing (NLP)

Natural language processing, commonly referred to as NLP, is a branch of artificial intelligence (AI) that deals with the interaction between computers and people using natural language. The NLP is need to apply algorithms to identify and extract the natural language rules with every sentence and collect basic data from it in order to convert unstructured language data into a form that the computer can understand.

Many useful and powerful libraries can apply natural language processing features to the project, such as spacy, Natural Language ToolKit (NLTK).

Spacy

The spaCy is a free open source library for NLP in Python, with many built-in functions. It is good at handling incredible large-scale information extraction tasks. Compared with other libraries, spaCy is a relatively young library, which is for production use only. It is the fastest parser in the market today. This is because the spaCy toolkit is written in Cython. This mean the spaCy can be process the text document faster than other library. SpaCy's features include non-destructive tokenization, named entity recognition, support for more than 59 languages, part-of-speech tagging, lemmatization and more. However, the spaCy is not support the stemming function.

Natural Language ToolKit (NLTK)

NLTK is one of Python's leading and best natural language processing libraries. It is the main tool for natural language processing and machine learning. Today, it has become the educational foundation for Python developers. NLTK is an essential library. It provides many tasks in Python, such as classification, stemming, tagging, parsing, semantic reasoning, and tokenization. This library is widely used in public, it has more than 100 corpora and related vocabulary resources, such as WordNet, Web Text corpus, NPS Chat, SemCor, FrameNet, etc. However, it may be process slower, unable to meet the needs of the rapid use of production.

2.2.9 Text Processing

Before perform the sentiment analysis on the social media comments, the natural language processing (NLP) step will be carry out noise removal, tokenization, stemming, lemmatization, part-of-speech tagging, and lowering the case of letters to mining the text. Therefore, the processed text can be further process using the machine learning (S. et al., 2018).

Tokenization

Tokenization is process to breaks sentences into tokens. It is the first process step of the natural language processing and corpus generation. It contains words and other elements, such as URL links, the common separator for recognizing a single word is a space, but other symbols can also be used. The tokenization of social media comments is more difficult than that of general text. This is because comments may contain many irrelevant words, such as punctuation marks, symbols, stop words, emoji, and specific characters. Therefore, during this process can filter the irrelevant token in text. After remove the stop words from the text, such 'is', 'a', 'an', 'the' and other, It make text look clearly and no dirty. The remove stop word can help to remove the extra common words from the sentences and also reduce its size in order to easy to identify the key word in the sentences and frequency distribution of concept word in the overall sentences. However, in some sentiment analysis research, it state that the remove of stop word is not necessary when apply the supervised machine learning to test classification (B.Chrystal and Joseph, 2015).

Noise Removal

Noise removal is removing the noise in the text. The noise is any part of the text document that does not add meaning and information to the data. For example of noise, hyperlinks, HTML tags, user mention in tweet. The hyperlinks or URL in the text processing would not add any value to the sentiment analysis. To remove the HTML tags from the text and only target the text. These twitter username are preceded by a '@' symbol, which does not convey any meaning. Therefore, these noises can remove.

Lowering the Case of Letter

Lowering the case of letter to make all token into lowercase. Lowering the case if the letter is most important in cleaning data process, this is because the without the lowercase the letter or word, the system will be consider two different thing even there are same word. This means that it is helpful and affect performance of model trained.

Stemming

There are various stemming strategies developed for different purposes. Stemming is referred to a crude heuristic process that used for chops off the beginning or end of the word, which is no identity to the morphological root of the word. In order the natural language processing (NLP) to perform the stemming function are require to use the natural language toolkit (NLTK) library but the spaCy library is not support for stemming function. Many stemming algorithms, whose purpose is to improve information retrieval performance, do not use a stem dictionary, but an explicit list of suffixes, and the criteria for removing suffixes. Stemmers of the perhaps most popular stemming tool, Porter stemmers, and Lancaster stemmers. The Porter stemmers was oldest developed in 1979, the Lancaster stemmers was developed in 1990, and it uses a more aggressive approach than Porter stemmers (Kolchyna et al., 2015).

Lemmatization

As opposed to stemming function, the lemmatization does not simply chop off inflections. Instead, it uses lexical knowledge bases to get the correct base forms of words. Lemmatization function is the process of get the 'lemma' of a word which involves reduce the word form after it understanding the part-of-speech (POS) and context in the sentences. The lemmatization function can be support by many NLP library such as NLTK, Spacy, TextBlob and other library. Lemmatization uses lexica to identify all possible lexical representations in the form of words. Rules are needed to express the allowable relationship between lexical and surface representations. The lemmatization can help slight improve precision and recall in the text classification. However, the lemmatization is slowest process compared with stemming function (Kolchyna et al., 2015).

Part-Of-Speech Tagging

The part-of-speech (POS) tagging can automatically tag each word of text in terms of which part of speech it belongs to a noun, pronoun, adverb, adjective, verb, interjection, intensifier, and other (Shubham et al., 2017). The goal of POS tagging is to be able to extract patterns from analysing frequency distributions of these POS tags and use them in the classification process as a feature (Shubham et al., 2017).

2.2.10 Six Basic Emotion Model

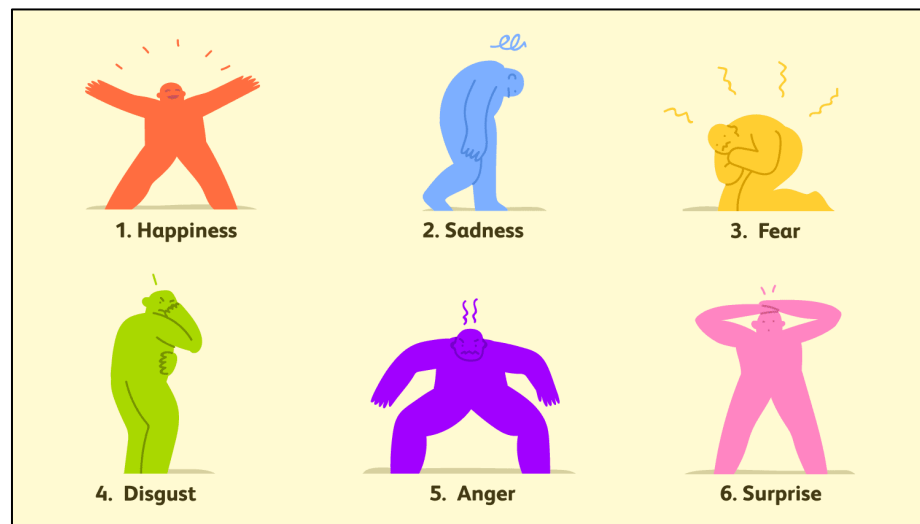


Figure 2.3: Type of basic emotions

The initial question of sentiment analysis is what emotions are present, and that will determine the classification classes.

Everyone has recognizable basic innate emotions. Basic emotions are biological emotions with unique universal signals. They are primitive emotions that a person will first unconsciously feel when any event occurs (Jain and Asawa, 2015). In the 1972's, psychologist Paul Ekman identified the basic emotions. He concluded that the six basic emotions are anger, disgust, fear, happiness, sadness, and surprise as shown in figure 2.2 (Balan et al., 2019). The basic emotions model also known as Ekman model. However, psychologist Paul Ekman explains that each emotion has its own specific characteristics, and can be express in varying degrees out. Each emotion acts as a discrete category rather than a separate emotional state.

Psychologist Paul Ekman initially considered six basic emotions. However, other theories and new research continue to explore other different type of emotions and how they to classified. Therefore, Psychologist Paul Ekman added some other emotions to the list, which are amusement, contempt, contentment, embarrassment, excitement, guilt, and pride in achievement, relief, satisfaction, and shame.

Anger can be a particularly powerful emotion defined as feelings of hostility, fury, frustration, and antagonism towards others. Anger can play a role in the body's fight or flight response. When there is a fight or flight reaction, the heart rate and breathing rate increase. Moreover, muscles become tense in response to threats in the environment. Disgust is synonymous with distaste, dislike, or repugnant. It may be derive from many things, including unpleasant taste, sight, or smell. Fear occurs when sensory organs sense pressure or dangerous stimuli. In addition, will experience fighting or escape reactions. Happiness often defined as a pleasant emotional state, synonymous with satisfaction, contentment, joy, and well-being. While sadness is the opposite of happiness, characterized by grief, disappointment, and pain. Surprise is usually quite brief and is cause by unexpected results, from amazement to shock.

The six basic emotions are core emotions that are universal throughout cultures all over the world and these are just a portion of the many different types of emotions. People can transmit their emotions through the surrounding mentioned emotion keywords and language attitudes (Lee and Chau, 2018). In addition, found that the Ekman's basic emotion set, arguably the most frequently used for classifying emotions.

2.2.11 Multiword Expression

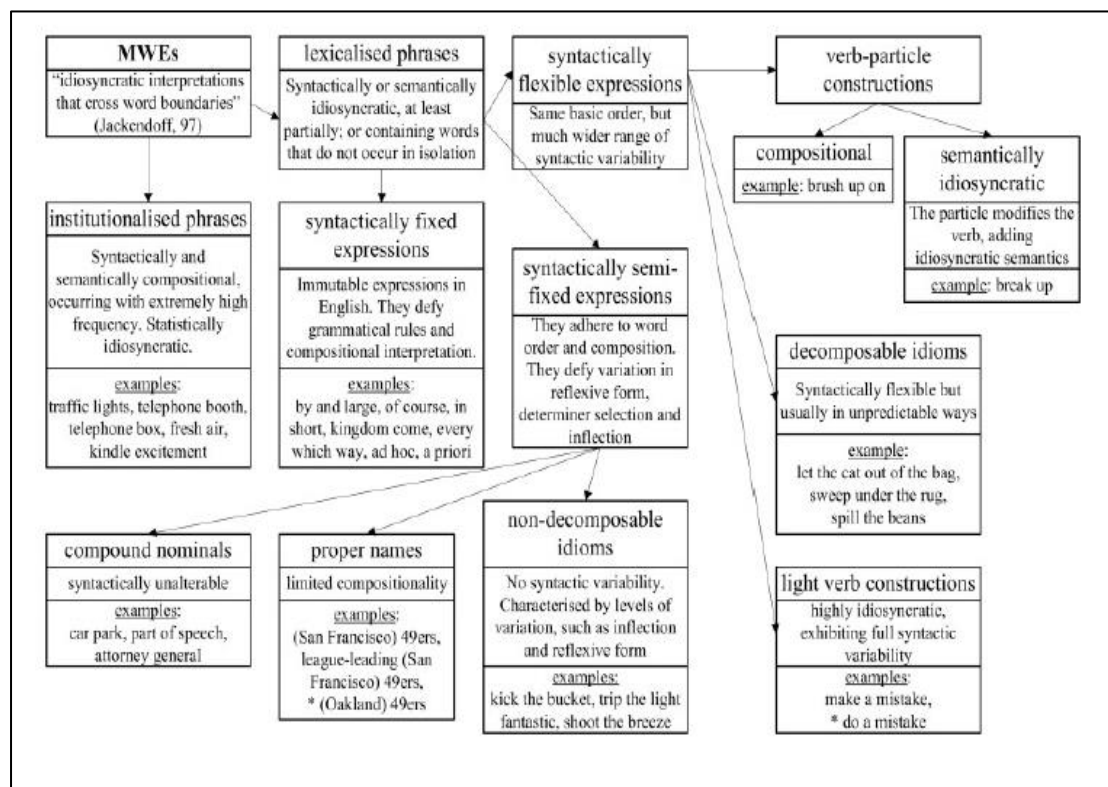


Figure 2.4: Type of Multiword Expressions

Traditionally, the sentiment analysis rely on pre-compiled dictionaries with sentiment or opinion words to predict the sentiment polarity of target sentences. Finding methods to represent the sentiment orientation of word sequences is an active research area of sentiment analysis. As researchers focus their research efforts on finding methods for identifying semantic orientations of single words or long documents. They may overlook the importance of Multiword Expressions, as we all know that the Multiword Expressions contain more semantic orientation information than words unit. So far, the Multiword Expressions are usually found in sentences, which means it sure to be more helpful for identifying semantic orientations of the sentences and even the document of speakers. Therefore, the purpose of this project is to propose three model to determine whether the performance of sentiment analysis can be improved.

Multiword Expressions (MWEs) are expression whose language behaviour cannot be predicted by the language behaviour of its constituent words. MWEs also known as verb-noun constructions (VNC). Generally, in the literature, MWEs are referred to as a combination of multiword units or words that appear more statistically than

opportunities. MWEs is a cover word for various types of collocations, and the transparency and fixity of these collocations vary. MWE is ubiquitous in natural language, especially in Web-based text and speech types. Therefore, can be considered that Multiword Expressions are crucial for any Natural Language Processing (NLP) task, this is because they often appear in any natural language. Recognizing MWEs and understanding their meaning is essential to understanding languages, so they are essential for any natural language processing (NLP) application that is designed to handle reliable language meanings and uses. The issue of MWE processing is the core problem of natural language processing (NLP). In natural language processing (NLP), it has caused many basic problems, and the frequency of these problems cannot be ignored (Constant et al., 2017).

There are many resources for major languages, such as English, Hindi and other languages, and there are many annotated corpora available in the field of the NLP. However, when it comes to the area of resource-scarce Indian languages, such as Bhojpuri, Magahi, Awadhi, Angika, Bagheli, Bundeli, Sadri, and many other tribal languages, there is no machine-readable text corpus available. This is because they are rarely recognized, resources are scarce and research is insufficient, so there is little contact with the technical field. Not only that, those languages and the more or less limited scope of view also will determine the complexity of their description and identification. (Kumar, Behera and Jha, 2017).

Multiword Expressions are expression whose linguistic behaviour is not predictable from the sentences or component words. they are also referred to as the combination of words or expressions that exhibit various linguistic oriented special features The characterizes of the Multiword Expressions are lack of compositionally manifest at the different levels of analysis, such as lexical, morphological, syntactic, statistical, and semantic (Salehi, Cook and Baldwin, 2015).

In general, the MWEs are not productive and they are flexible, though they allow for inflectional variation. MWEs can be defined as a word collocation that refers to a single concept. MWEs as a unit can be predicted from the meaning of its constituent words, for example, "make a decision" means "decide." And this can also be well explained by the expression "by and large", any English user knows that its adverb meaning and

syntactic function are almost the same as "mostly", an adverb. Some type of MWEs are more predictable than other constructions, for example, 'kick the bucket' when used idiomatically to mean 'to die' (Constant et al., 2017).

Nowadays, most research has addressed the problem of MWEs type in English. It stated that the MWEs are classified into various type is based on their lexical and semantic characteristics. This mean that MWEs are not homogeneous and have been categorized using different schemes. The figure 2.4 is showing the different type of MWEs, it is consists the fixed expression, light-verb construction, verb-particle construction, nominal compounds, decomposable idioms, non-decomposable idioms and so on (Constant et al., 2017). There are example of subtype of Multiword Expression.

- Fixed expressions: 'kingdom come', 'ad hoc'.
- Light Verb Constructions: 'make a living', 'make a decision', 'take a break'.
- Verb Particle Constructions: 'write-up', 'call-up', 'add in'.
- Idioms: 'spill the beans', 'kick the bucket', 'pop the question'.

Although their actual definitions are different, in its widest meaning reaching proper names or fixed idiom expressions (with non-constitutive meanings) to light verb construction or seemingly free, but they are actually statistically idiosyncratic allocations, therefore the different applications and implementations also can deal with them.

It is important to recognize expressions and classifications that represent emotions or emotional states in text data. In this regard, MWE can not be overlooked because they constitute a significant proportion of the emotion lexicon, as it implies that the multiword expression itself may often be sufficient to determine the underlying sentiment. (Fotopoulou, Aggeliki & Giouli, Voula, 2018).

The most important part of the multiword Expression is to be able to differentiate between it and literal combinations that have the same surface. For example: ' he spilt the beans on the kitchen counter' is most likely a literal usage, based on the sentence, it is given away by the use of the prepositional phrase on the 'kitchen counter'. This is

because the beans could have literally spilt on the place of the kitchen counter. Based on the research, it points out that 53 idiom MWE types used in different contexts, and concluded that more than half are literal meaning. In addition, it shows the machine translator may have bad effects on the quality of the sentences (Salehi, Cook and Baldwin, 2015).

The Multiword Expressions also important in NLP application, such as the machine translation and it is widely recognized. This is because that most of Multiword Expressions cannot be translated literally, such as the 'dead loss', 'to make an appointment', 'to kick the bucket'. Not only that, some of the compounds in the Multiword Expressions, as well as some fixed expressions, do not respect grammatical rules, for example 'by and large' (Wehrli, 2017).

As mentioned earlier, therefore, most NLP applications need to "understand" and correctly recognize MWE. However, if you consider the syntactic flexibility of many MWEs, this may become very complicated, and this is limited to the following cases. Adjectives or adverb modifiers can usually be added to Multiword Expressions to separate the two terms, such as 'a school of little fishes'. Several types of collocations can be grammatically processed, and these grammars may modify the standard order of collocations, such as passive, relativized, etc. Sometimes, nouns in verb-object or subject-verb collocations can be replaced by pronouns (Wehrli, 2017).

To date, Multiword Expressions have been studied for many years by researchers. In the research, the researches have solve the problem of the Multiword Expressions classification in English at the token-level. Recent work on token-level Multiword Expressions classification has focused on approaches that are applicable to the full range of type of Multiword Expressions. Multiword Expressions classification is the task of determining, at the token level, which words are Multiword Expressions, and which are not. In the research, they use the supervised machine learning approach to perform the Multiword Expressions classification. (Wehrli, 2017)

Most of the research, the supervised machine learning approach based on the structured perception or the unsupervised learning algorithm with the clustering approach to segment the corpus into multiword units based on the predictability are more frequency

use as approach to perform their research and achieve their goals. However, in literature review, the researchers proposed the first deep learning models for broad-coverage MWE identification, which are a layered feedforward network (LFN), a Recurrent Neural Network (RNN), and two convolutional neural network (CNN). In addition, they also compare these models against the state-of-the-art method and some traditional supervised machine learning approaches which are the k-Nearest Neighbour, Random Forest, Logistic Regression, and Gradient Boosting. They used the DiMSUM dataset for training and evaluating in order to get the performance. In order to improve the performance, they perform natural language processing (NLP) tasks. They perform the tokenization, amortization, Part-Of-Speech (POS) tag; and gold-standard Multiword Expressions (MWEs) tag. As a result, they show that their deep learning models are able to provide more effective and high performance than previous approaches, based on comparisons between the performance of models on validation and test data. (Gharbieh, Bhavsar and Cook, 2017).

In this project, is required to extract and detect the multiword expression in the sentence, are not perform the Multiword Expressions classification, therefore, in this project are not using the supervised machine learning approach, or the unsupervised machine learning approach or the deep learning model in the Multiword Expression dataset. Extracting Multiword Expression, might is an important task for the sentiment analysis of texts or feedback as the Multiword Expressions can express more integrative sentiments than words units. The efficient extract and identification of Multiword Expressions can need the Natural Language Processing (NLP) tasks such as parsing sentences to process. The Multiword Expressions may influence some NLP tasks and important research problem in the field of NLP (Farahmand & Nivre, 2015). Therefore, the deconstructing text into concepts is important to semantic perception analysis of the text. Concepts can be single words or Multiword Expressions. Some of latter are semantic atoms that can never be broken down into single words. For example, the concept of painkillers should not be divided into pain and killer, the semantics of these two words are completely different. The method of extracting Multiword Expressions is usually to use POS tagging and text chunking to extract the n-gram to be processed (Cambria et al., 2017). Even found that there are many types of Multiword Expressions, only the light verb construction will be discussed in this project.

2.2.12 Light Verb Construction

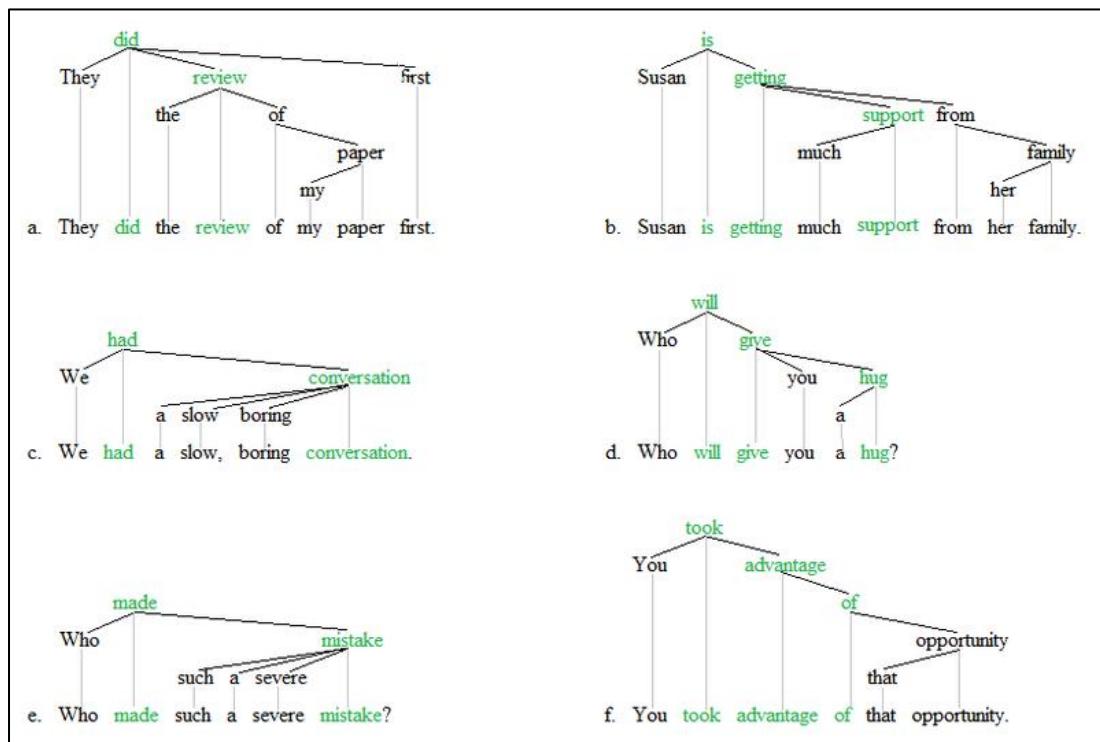


Figure 2.5: dependency grammar trees of the light verb construction

The light verb constructions, which are also known by various other names such as expanded predicate, verbs-nominal construction, stretched verb constructions, or support verb constructions. It is the most semantically common, so it is a combination of very frequent verbs and predicate nouns that usually represent language processes (Ronan and Schneider, 2015).

A lot of theoretical research has been conducted on the use of light verb structures in different languages, including English, Japanese, Korean and other languages (Vaidya, Agarwal and Palmer, 2016). Think of light verb constructions as those structures, which are collocations of an inflexible verb.

The light verb constructions were difficult to present, this is because some of them were not compositional in a straightforward way. The light verb construction was flexible. This makes the light verb construction become very complicated due to the widespread use of adjectives and adverb. This fact is obvious, this is because the words that make up the light verb structure usually don't have any constituent elements in any sense.

However, these constructions do form of chains. The figure 2.5 shows the dependency grammar trees of the light verb construction. Each word constructed by a light verb forms a conjunction. In this regard, the green word becomes the main predicate of the clause every time. If there is an auxiliary verb (in examples b and d), the auxiliary verb is included in the subject-predicate because, like the light verb, it only contributes to the functional meaning.

The light verb constructions, such as 'make an offer' and 'give a groan,' are considered to consist of semantically common verbs and nouns that represent events or states. The light verb construction usually has low semantic specificity, with predicate nouns, and in many cases action nouns. These collocations can usually be explained with a simple verb. For example of a structure is whether to 'make suggestions' or 'suggest', and whether the 'give an example' versus to 'exemplify' and 'make a decision' to 'decide' (Ronan and Schneider, 2015).

The most 6 common found verbs are the semantically most general ones, which are 'do', 'get', 'give', 'have', 'make', and 'take'. Example for each of them, 'do a bunk', 'get a move on', 'give a hug', 'have a look', 'make a plunge' and 'take a walk'. Due to their general applicability are the best verbs in contemporary English in general, but they can also be found more semantically specific verb, (Ronan and Schneider, 2015). Moreover, conceptually, these words have a more primitive predicate meaning than other verbs. In fact, these verbs remind people of conceptual categories, such as primitive predicates in the conceptual structure of words (Bak, n.d.).

Interestingly, the most common verb "do" in English may not be the most productive prone verbs in idioms. This can be explained by the fact that although the verbs in the sentence usually have a general meaning, they are usually the main verbs. On the other hand, "do" has been increasing its use as an auxiliary word in the past few centuries and has lost its status as the main verb. This means that the "do" verb becomes the common verb usage in the light verb construction. Although it can be done in English, like 'do an about-face' or 'do one's head in indicate', but it is rare. Therefore, can concluded that there are two types of "do" in light verb construction English: main verbs and auxiliary verb (Nenonen et al, 2017).

The light verb construction is formed from a commonly used a verb and a noun phase in the direct object position, it mean that light verb construction = verb + noun / V+NP. However, analysis the light verb is not simple task, it is a more complex verb predicates do not clearly discrete into binary distinction of compositional or non-compositional expression. For example: 'take a bag', 'take a while', and 'take a walk'. These three constructions are cosmetically very similar, but look at the semantics of three constructions, reveals significant different and each of them are showing that a different type of MWEs. The first expression 'take a bag' is only the literal combination of a verb and an object noun, it is not consider is light verb construction. The second expression 'take a while' is a type of MWEs but it is consider category into idiom, not a light verb construction. It is meaning a task may take some time to complete. Only the last expression 'take a walk' is a light verb constructions whose meaning mainly derives from its component, which is 'walk' object noun, while the meaning of the main verb 'walk' is somewhat bleached.

The light verb construction have already been identified as one of the major sources of problems in various Natural Language Processing (NLP) task. Such as the automatic word alignment and semantic annotation transference. These problems are provide an empirical basis for differentiation between the bleached and full meaning of a full verb within a given text (Ronan and Schneider, 2015).

In addition, there was a research that was related to the classification. The research describes the development of a supervised machine learning approach of English light verb constructions. This approach has relied on the features from dependency parses, OntoNotes sense tags, WordNet hypernyms and WordNet lexical file information. In the research, the researcher compared the two different light verb construction dataset resources, which are dataset from e British National Corpus (BNC) produced by the Tu and Roth and the dataset from the Prop Bank (PB). in the comparison, the light verb construction dataset from the BNC was selected this is because the dataset consists of 2,162 English sentences and involve the six more frequent light verbs, which are 'do', 'get', 'give', 'have', 'make', and 'take'. Another reason is the dataset from PB was loosening the annotation requirements and including such semi-light usages in the light verb constructions annotations. As a result, researchers show that the WordNet

hypernyms classifier provided the high performance on the F1 score. (Chen, Bonial and Palmer, 2015)

Furthermore, the light verb construction has also been studied in different English domains. The research was determining the light verb constructions in contemporary British English and Irish English. It was a study about implementing an automatic parser-based approach to investigate the light verb constructions. In their research, two different datasets consist of the British English and Irish English was used. In the research, they explain and evaluate the steps employed to optimize parser output in identify open lists of light verb constructions. Not only that, but they also discuss the usage of two different data, this is because of their structures are different. They also found that if the search is not limited to certain high-frequency light verbs, the light verb constructions will be greatly increased. Therefore, they using manual or semi-automatic approaches to data collection. They were used the T-score collocation measure, NomLex and Wordnet to perform the parser in two datasets to detect the overall amount of light verb construction. In the study, those approaches they used neither limited the number of allowed light verbs nor the morphological pattern of predicate nouns. As a result, found that the T-score collocation measure seems better than other approaches (Ronan and Schneider, 2015).

Based on various research, most researches were more focus on classifying the light verb construction in sentences to provide the binary result (+ and -), which is '+' is the light verb construction, while '-' is the non-light verb construction. They are most using the dataset in British National Corpus (BNC) produced Tu and Roth. Not only that, but some researches have also used the parser-based approach in the light verb construction, the parser-based approach relies on the part-of-speech (POS) tagging, which is Natural language processing task, in order to understand the meaning of a sentence or word to provide the sentence parser tree. However, unfortunately, very rare researches discuss the light verb construction related to the sentiment analysis.

Moreover, some time the light verb construction was consist of the adjective and adverb, such as 'make a nice decision', but some researches are not consider about it, this is because most of them are perform the classification project while less perform extract or detecting the light verb construction. In this project, the light verb

construction is core of project, the sentences may consist of light verb construction, therefore needs to be explicitly encoded in order to detect and extract the light verb construction and replace with a same meaning single word to simply the sentence to make it more shorter, so the model can be easy to target the sentiment orientation of the word in the sentiment analysis. This project also propose to perform the comparison to determine whether the extraction of light verb constructions can help to improve the sentiment analysis. Therefore, there are two requirements for light verb construction to be effectively utilized in sentiment analysis model: first, light verb construction need to be recognized in text or sentences, second, need the explicitly encode to detect and extract the light verb constructions. In this project, extracting the light verb construction approach was using the spaCy library to perform the rule-based matching approach to extract the light verb construction from the sentences.

2.2.13 Cross-Industry Standard Process for Data Mining

The Cross-Industry Standard Process for Data Mining (CRISP-DM) project proposes a comprehensive process model for executing data mining project. This process model has independent of the industry and the technology used. Data mining is a creative process that requires several different skills and knowledge. The data mining requires a standard method that will help transform business problems into data mining tasks, suggest appropriate data conversion and data mining techniques, and provide methods for evaluating the effectiveness of results and recording experience. The CRISP-DM model can be used as a general reference point for discussing data mining and helps all participants understand the key data mining issues. The CRISP-DM model can provide a structured approach and an overview of the life cycle in order to plan a data mining in this project. The life cycle of the CRISP-DM is broken down in six phases, which is business understanding, data understanding, data preparation, modelling, evaluation and deployment (Rodrigues, I, 2020).

Business understanding phase is focusing on the understanding the project object, it is determine the objective, assess the current situation, determine the data mining goal and produce a project plan. Assessing the current situation requires a more detailed factual survey of all resources, constraints, assumptions and other factors. The data mining goal elaborates the project goal in technical terms. Then, to describe a plan for the data mining goal and thereby to achieve the project objective.

In data understanding phase requires to acquire the data listed in the project resources to understand what can be expected and achieved from the data. It consists of collect initial data, describe data, explore data, and verify data quality.

In data preparation phase is to decide and prepare the data is going to use for analysis. The tasks in this phase involve select data, clean data, construct data, and integrate data. Then covers all activities to build the final dataset as structure data file.

In modeling phase is responsible for the results that should meet the project objective. It is consist tasks, such as select the actual modelling technique, generate test design,

build model, and assess model. The technique was selected is support vector machine (SVM), Naïve Bayes, and Logistic Regression.

In the evaluation phase, for a more thorough evaluation of the model, it is important to check the steps performed to build the model to ensure that the model can correctly achieve the project objective. The tasks is consist of evaluate the results, review process, determine the next steps.

In the deployment phase as the last phase is to organize and present the project in a useful and understandable manner. The project will present in the web-based system. The tasks in deployment phases are consist of plan the deployment, plan monitoring and maintenance and produce a final report.

2.3 Chapter Summary and Evaluation

In this chapter, the project background has been discussed. The existence of light verb construction detection is aim to improve the performance measurement of sentiment analysis in term of accuracy, precision, recall and F1 score. In addition, this chapter also do the literature review such as sentiment analysis, sentiment analysis algorithm, machine learning, supervised machine learning, support vector machine (SVM), Naïve Bayes, Logistic Regression, text processing, six basic emotion model, Multiword Expressions (MWEs) and light verb construction. Based on literature review, the MWEs is will be key problem in sentiment analysis, and. The MWEs will process after the text processing and text process will be mining the data and remove some irrelevant word or character from the sentence, through this, the light verb construction detection can to detect the light verb in the sentence more successfully.

Chapter 3

Methodology and Requirements Analysis

3. Methodology and Requirements Analysis

The Cross-Industry Process for Data Mining (CRISP-DM) methodology will be used in this project. The CRISP-DM methodology will provide a structured approach to planning a sentiment analysis by detecting light verb construction. This model has an idealized sequence of tasks to perform and evaluate the data mining. It consists of 6 major stages to conceive a sentiment analysis by detecting light verb construction project. Those stages are Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation and the Deployment. Each of them consists of a few tasks as shown in table 3.1

Stage	Task
Business Understanding	determine data goals, produce project plan
Data Understanding	collect initial data, describe data, explore data, verify data quality
Data Preparation	select data, clean data, format data
Modeling	select modeling technique, generate test design, build model, assess model
Evaluation	evaluate the result, review of process
Deployment	plan deployment

Table 3.1: stages and tasks in CRISP-DM methodology

3.1 Business Understanding

The first stage of CRISP-DM process is understanding the goals and requirement of this project. It is consist a few task. Which are determine data mining goals and produce project plan.

3.1.1 Determine Data Mining Goals

Data mining Goals

There are a few objective in this project:

- To identify the sentiment from the sentences.
- To detect the light verb construction in sentences.
- To improve accuracy of sentiment analysis by detecting the light verb construction.
- To compare the performance of sentiment analysis algorithms.

Data mining success criteria

- The accuracy of sentiment analysis after detecting light verb construction higher than the accuracy of sentiment analysis without detecting light verb construction.
- The model can produce higher precision and recall.

3.1.2 Produce Project Plan

The project plan prepare is in order to make sure to achieve the project object, which describe in chapter 1.4.

3.2 Data Understanding

Second stage of the CRISP-DM process to understand the expected and achieved from the data. There are a few tasks in this stage, which are collect initial data, describe data, explore data, and verify data quality.

3.2.1 Collect Initial Data

Data collection is important task, it is use to make sure can successfully achieve the data mining goals. In this process of this project is require two different data source, which are light verb construction dataset and sentiment analysis dataset.

Light Verb Construction Dataset

The light verb construction dataset come from the British National Corpus (BNC) and it was created by Tu and Roth. The dataset has contains of 2,162 English sentences, and involve the six most common light verbs: ‘do’, ‘get’, ‘give’, ‘have’, ‘make’, and ‘take’. Tu and Roth are used this data set to study the distinction the light verb constructions between non-light verb constructions by train the machine learning algorithm classifier. Their dataset was used for the machine learning algorithm train, therefore, the dataset resource was contained some overlapping and distinct constructions. The Light verb construction dataset created by Tu and Roth has high frequency used by other researchers and it is get from the link below.
http://multiword.sourceforge.net/PHITE.php%3Fsitesig%3DFILES%26page%3DFILES_20_Data_Sets

Sentiment Analysis Dataset

The sentiment analysis dataset is used for the supervised machine learning algorithm to train and classifier the sentiment in the sentences. This dataset is important for this project, this is because it can make sure the project can be successfully to achieve the objective and goals. The sentiment analysis dataset is different the common dataset for using sentiment analysis. The sentiment analysis dataset in this project are require 6 label classes, which are anger, sad, joy, fear, disgust and surprise.

3.2.2 Describe Data

Light Verb construction Dataset

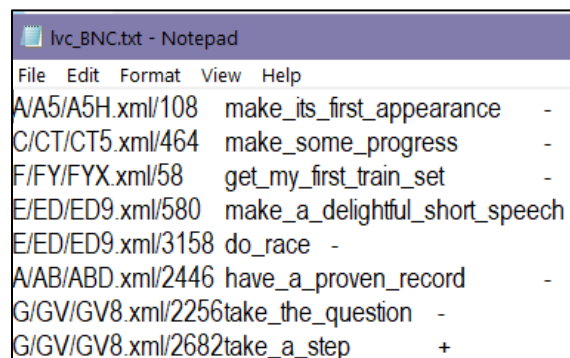


Figure 3.1: light verb construction dataset

- File Format: Text file format
- Shape of data: 2,162 rows, 3 columns

Data	Description	Example
ID	Unique id to identify the records	A/A6/A6F.xml/281
Sentences	The sentences is contain light verb constructions and non-light verb construction.	take_the_question take_a_step
Outcome	'+' represent light verb construction '-' represent non-light verb construction	'+', '-'

Table 3.2: data detail of light verb construction dataset

Sentiment Analysis Dataset

emotion	sentence
anger	I've got a bone to pick with one club
anger	Get off my back
anger	I hate it
anger	Alain was in a black mood and Marguerite was trying hard to pretend she wanted to
anger	The frustration and worry she felt at the moment were enough to make her want to

Figure 3.2: sentiment analysis dataset

File format: csv file format

Shape of data: 18,419 rows and 2 columns

Missing value: 0 missing value

Data	Description
Emotion	It is contain 6 different label class and it is outcome/ Y-axis.
Sentences	Text format, it is tweet from the twitter and it is feature / X-axis.

Table 3.3: data detail of sentiment analysis dataset

3.2.3 Explore Data

Light Verb Construction Dataset

- Six most common light verbs (: ‘do’, ‘get’, ‘give’, ‘have’, ‘make’, ‘take’)

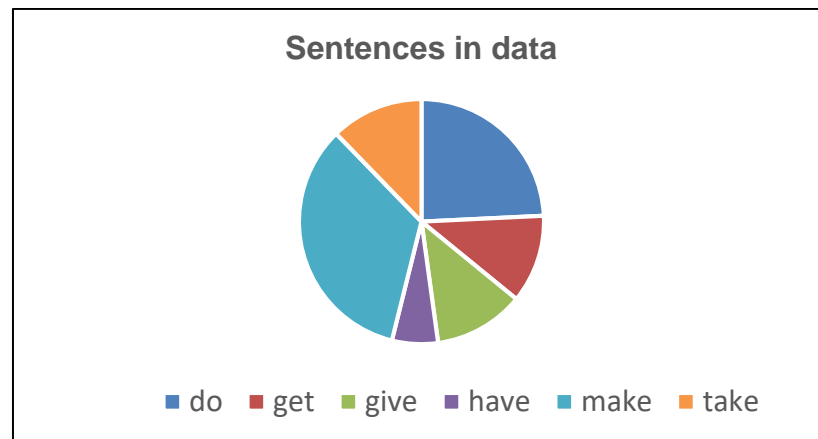


Figure 3.3: Pie chart of common light verbs

- ‘do’ light verbs: 524 records
- ‘get’ light verbs: 252 records
- ‘give’ light verbs: 258 records
- ‘have’ light verbs: 131 records
- ‘make’ light verbs: 732 records
- ‘take’ light verbs: 265 records
- Outcome quality

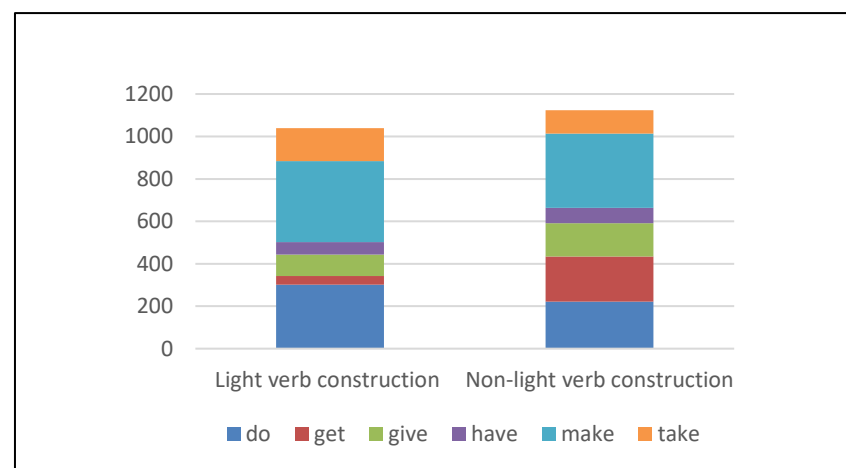


Figure 3.4: number of light verb construction and non-light verb construction

Outcome: number of ‘+’ data (light verb construction): 1039 records

Outcome: number of ‘-’ data (non-light verb construction): 1123 records

Sentiment Analysis Dataset

- 6 emotion in data

The 6 emotion is based on the Ekman model, it consist of anger, disgust, fear, happiness, sadness, and surprise, each of them have their specific characteritics.

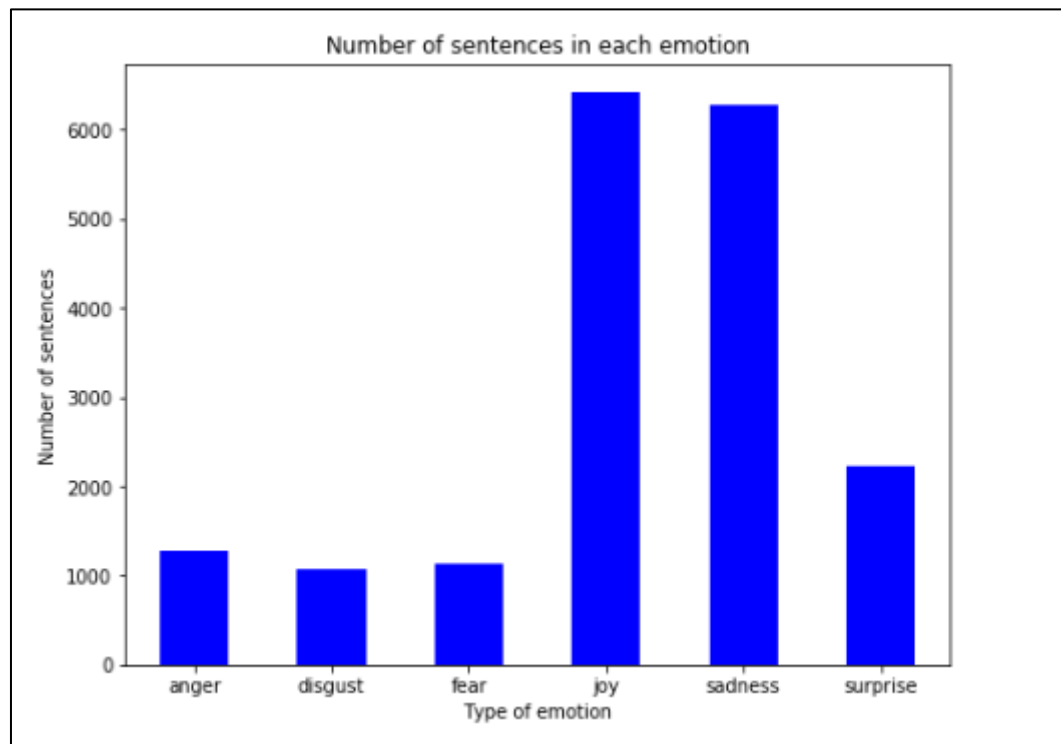


Figure 3.5: number of sentences in each emotions

Joy:	6,417 sentences (34.84%)
Sadness:	6,280 sentences (34.10%)
Surprise:	2,227 sentences (12.09%)
Anger	1,278 sentences (6.94%)
Fear:	1,140 sentences (6.19%)
Disgust:	1,077 sentences (5.85%)

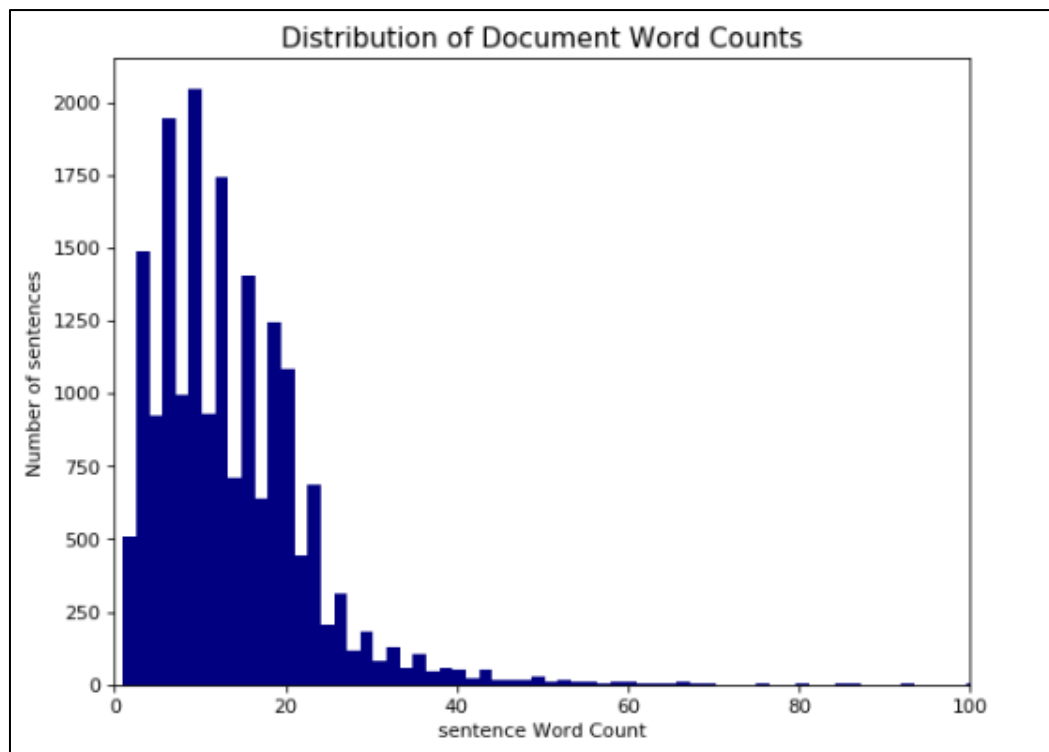


Figure 3.6: distribution of word counts in sentiment analysis dataset

The Figure 3.6 is showing the bar chart of the distribution of word counts in the sentiment analysis dataset, it can present more of the word counts in sentences is between the 1 and 20. However, some of the sentences is more then 60 word, it is too long in the dataset.

3.2.4 Verify Data Quality

Light verb Construction

The light verb construction dataset is incomplete for used in the project. In this project, are require to replace the light verb construction to full verb with same meaning, example 'make a decision' replace to 'decide'. In order solve this problem, can use python to perform the natural language processing (NLP) function to convert noun 'decision' to verb 'decide' with using the WordNet corpus. At the same time, the non-light verb construction sentences is irrelevant and unnecessary in this project, therefor it is require to filter up. In addition, the dataset was contain many duplicate data and the sentence are consist the underscore such as 'do_the_work'.

Sentiment Analysis Dataset

The sentiment analysis dataset is complete and correct. This mean that it suitable use to perform the classification. However, the sentence in the dataset is consist many noise, therefore, the sentences in dataset require to preprocessing.

3.3 Data Preparation

The data preparation involve the tasks in order to turn o piece of data into useful by the algorithm and process. The task are select data, clean data, and format data.

3.3.1 Select Data

Light Verb Construction Dataset

The light verb construction dataset will be used for in the detecting the light verb construction function, it is to make sure the sentences extract by the approach is matching, it is avoid some construction similar to light verb construction but it is non-light verb construction. Therefore the dataset will be filter the non-light verb construction. And remove irrelevant data variable, which are id and outcome, only the light verb construction sentences.

Sentiment Analysis Dataset

The sentiment analysis dataset is complete, each of them is important used for train the model. The sentences data as the feature / x-axis and the sentiment as outcome / y-axis. However, the sentences in the dataset require to process text processing before go to model train.

3.3.2 Clean Data

Light Verb Construction Dataset

- Remove the non-light verb construction
- Remove the id and outcome
- Remove the underscore '_'
- Lower case the letter

The light verb construction is used for extract the light verb construction from the sentence then to check whether it is match with data in the dataset. The outcome of light verb construction is '+', then only take import csv to open the text file and use the list only consist of the light verb construction sentences. Then the light verb construction sentences is consist the underscore, therefor the underscore of text is require to remove. Then to process the tokenization to only contain the alpha token and lower case the token. The coding is shown in figure 3.7.

```
lvclist=[]
with open('lvc_BNC.txt') as txtfile:
    readtxt = csv.reader(txtfile, delimiter='\t')
    for row in readtxt:
        if row[2] == '+':
            str = row[1].replace('_', ' ')
            sentence = nlp(str)
            text = [token.text.strip().lower() for token in sentence if token.is_alpha == True]
            text1=' '.join(text)
            lvclist.append([text1])
```

Figure 3.7: coding of clean light verb construction dataset 1

- Remove the duplicate data
- Sorted the data

The light verb construction sentence in the dataset found that there are many overlapping and duplicate there to remove the duplicate data and sorted the data to according sequence of 'do', 'get', 'give', 'have', 'make', and 'take'. The coding as shown in figure 3.8.

```
# removing duplicate data from the nested list
lvclist = list(set(map(lambda i: tuple(sorted(i)), lvclist)))
#sort the data from nested list
lvclist=(sorted(lvclist, key=operator.itemgetter(0)))
```

Figure 3.8: coding of clean light verb construction dataset 2

- Add the convert noun

The light verb construction sentence using WordNet corpus to convert the noun in the sentence to the verb. Each of light verb construction sentences is need process coding shown in figure 3.9, the code will convert to get the result and return the possible verb, and only take the high probability.

```
def convert(word):
    WN_ADJECTIVE = 'a'
    WN_ADJECTIVE_SATELLITE = 's'
    synsets = wn.synsets(word, pos='n')
    # Word not found
    if not synsets:
        return []
    # Get all lemmas of the word (consider 'a' and 's' equivalent)
    lemmas = []
    for s in synsets:
        for l in s.lemmas():
            if s.name().split('.')[1] == 'n' or 'n' in (WN_ADJECTIVE, WN_ADJECTIVE_SATELLITE) and s.name().split('.')[1] in (WN_ADJECTIVE, WN_ADJECTIVE_SATELLITE):
                lemmas += [l]
    # Get related forms
    derivationally_related_forms = [(l, l.derivationally_related_forms()) for l in lemmas]
    # filter only the desired pos (consider 'a' and 's' equivalent)
    related_noun_lemmas = []
    for drf in derivationally_related_forms:
        for l in drf[1]:
            if l.synset().name().split('.')[1] == 'v' or 'v' in (WN_ADJECTIVE, WN_ADJECTIVE_SATELLITE) and l.synset().name().split('.')[1] in (WN_ADJECTIVE, WN_ADJECTIVE_SATELLITE):
                related_noun_lemmas += [l]
    # Extract the words from the lemmas
    words = [l.name() for l in related_noun_lemmas]
    len_words = len(words)
    # Build the result in the form of a list containing tuples (word, probability)
    result = [(w, float(words.count(w)) / len_words) for w in set(words)]
    result.sort(key=lambda w: w[1])
    # return all the possibilities sorted by probability
    return result
```

Figure 3.9: coding of clean light verb construction dataset 3

The light verb dataset after process the clean data, there are 760 unique light verb construction sentences and the result of the dataset will shown in chapter 3.3.3 format data.

Sentiment Analysis Dataset

- Remove URL link
- Remove HTML tag
- Remove @username
- Remove accents
- Remove punctuation
- Replace emoji to word

The sentence in the sentiment analysis dataset is consist many irrelevant data or word. Therefore require to process clean. For example, the URL link, html tag, @username, accent, punctuation, emoji is not contribute to the sentiment analysis. The coding as shown in figure 3.10

```
def clean_url(text):
    cleanr = re.compile('(https?:\\/\\/)(\\s)*(www\\.)?(\\s)*((\\w|\\s)+\\.)*([\\w\\-\\s]+\\/)*([\\w\\-]+)((\\?)?([\\w\\s]*=\\s*[\\w\\%&]*\\s*))')
    cleantext = re.sub(cleanr, '', text)
    return cleantext

def clean_html(text):
    cleanr = re.compile('<.*?>|&([a-z0-9]+|#[0-9]{1,6}|#x[0-9a-f]{1,6});')
    cleantext = re.sub(cleanr, '', text)
    return cleantext

def clean_username(text):
    cleanr = re.compile('@([\\s]+)')
    cleantext = re.sub(cleanr, '', text)
    return cleantext

def strip_accents(text):
    return ''.join(c for c in unicodedata.normalize('NFD', text) if not unicodedata.combining(c))

def clean_punc(text):
    punctuation = "!@#$%^&*()_+=[]{}:|\\/<>?:.-,;"
    for c in text:
        if c in punctuation:
            text = text.replace(c, ' ')
    return text

def replace_emoji(text):
    return emoji.demojize(text, delimiters=(" ", " "))
```

Figure 3.10: coding of clean sentiment analysis dataset 1

- Convert to lowercase
- Remove stop words
- Tokenization
- Apply lemmatization or Stemming
- Remove the digit/number

Then the sentences to process the lower case the word. However, the remove stop word is problem, this is because if the sentence are remove stop word, such as 'a', 'an', 'the', this will make the sentence cannot extract or detect the light verb construction in the sentences. This mean that, if remove the stop word, it will remove some important word lose and affect this project goal. Therefore, in this project is not remove the stop word. At the same time, there are two way to help remove sparse terms and particular words, there are stemming and lemmatization function. The stemming and lemmatization can help to identify the multiword expression in data, this is because the third person form, present tense and past tense of word are hard to detect the multiword expression. Such as 'made a decisions', therefore use stemming or lemmatization to become 'make a decision'. However, there are different, therefore need to analysis to determine whether stemming or lemmatization is most suitable in this project for clean data.

Stemming

In order the natural language processing (NLP) to perform the stemming function are require to use the natural language toolkit (NLTK) library but the spaCy library is not support for stemming function. The NLTK can be choose the Porter Stemmer or Lancaster Stemmer to perform the stemming function. The Porter Stemmer was oldest developed in 1979 and the Lancaster Stemmer was developed in 1990 and it uses a more aggressive approach than Porter Stemmer. The below will be demo the Porter Stemmer and Lancaster Stemmer to see how their performance.

Porter Stemmer

```
from nltk.stem import PorterStemmer
ps=PorterStemmer()
words = ["caring", "studies", "better", "languages"]
print('')
print ('Posster Stemmer')
for w in words:
    print(w, " : ", ps.stem(w))
```

Figure 3.11: code of Porter Stemmer

```
Posster Stemmer
caring : care
studies : studi
better : better
languages : languag
```

Figure 3.12: output of Porter Stemmer

Lancaster Stemmer

```
from nltk.stem import LancasterStemmer
ls = LancasterStemmer()
words = ["caring", "studies", "better", "languages"]
print('')
print ('Lancaster Stemmer')
for w in words:
    print(w, " : ", ls.stem(w))
```

Figure 3.13: code of Lancaster Stemmer

```
Lancaster Stemmer
caring : car
studies : study
better : bet
languages : langu
```

Figure 3.14: output of code

Lemmatization

The below will be perform the lemmatization function by using the spaCy library, which load en_core_web_sm library model and .NLTK library to load the WordNet library model. The WordNet is a large and publicly available lexical database for the English language aiming to establish structured semantic relationships between words. The WordNet offers lemmatization capabilities as well and is the earliest and most commonly used to perform lemmatization. spaCy is a relatively new in the space and is billed as an industrial-strength NLP engine. The spaCy is easy to use, this is because that it can parse text and compute various NLP related features through one single function call to perform the lemmatization.

Spacy Lemmatization:

```
import spacy
nlp = spacy.load('en_core_web_sm')
words = ["caring", "studies", "better", "languages"]
doc = nlp(' '.join(words))
print('')
print ('spaCy lemmatization')
for w in doc:
    print(w, " : ", w.lemma_)
```

Figure 3.15: code of spaCy lemmatization

```
spaCy lemmatization
caring : care
studies : study
better : well
languages : language
```

Figure 3.16: output of spaCy lemmatization function

WordNet Lemmatization:

```

from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
words = ["caring", "studies", "better", "languages"]
print('Wordnet lemmatization')
for w in words:
    print(w, ":", lemmatizer.lemmatize(w))

```

Figure 3.17: code of WordNet lemmatization

```

Wordnet lemmatization
caring : caring
studies : study
better : better
languages : language

```

Figure 3.18: output of WordNet lemmatization function

After collect the performance of stemming and lemmatization as shown in figure 3.11 to 3.18. The lemmatization function perform by spacy library is most suitable use in our project. This is because the stemming will often leading to incorrect meanings and spelling and the WordNet lemmatization is less powerful and slower than spacy library. The figure 3.19 is coding of clean data, process the lower case the word, lemmatization and remove number.

```

def tokenization(text):
    # put the comment into spacy model
    sentence = nlp(text)
    #data pre-processing
    #tokenization and Lemmatization and lowercase the text and remove digit
    text = [token.lemma_.strip().lower() for token in sentence if token.is_digit==False and token.is_alpha == True]
    #make it token become sentence
    sentence_text = ' '.join(text)
    return sentence_text

```

Figure 3.19: coding of clean sentiment analysis dataset 2.

- Extract the light verb construction and replace to full verb

The extract light verb construction from the sentences in the dataset is most important in this project. After analysis light verb construction, this project is using the rule-based matching to match the pattern (verb + noun). At the same, this project also consider the light verb construction consist of adjective and adverb, such as ‘take a long break’, ‘make a nice decision’. The rule-based matching to matching part-of-speech tagging to match pattern to extract the light verb construction from the sentence and match the dataset in light verb construction dataset to make sure it is correct light verb construction and replace with a single same meaning verb. The coding as shown in figure 3.20.

```

# Light verb detection
def extract_lvc(string, filename):
    nlp_doc = nlp(string)
    #rule based approach
    #declare the pattern
    pattern = [{ 'POS': 'VERB' }, { 'POS': 'DET', 'OP': '*' }, { 'POS': 'ADV', 'OP': '?' }, { 'POS': 'ADJ', 'OP': '*' }, { 'POS': 'NOUN', 'OP': '+' } ]
    pattern1 = [{ 'POS': 'AUX' }, { 'POS': 'DET', 'OP': '*' }, { 'POS': 'ADV', 'OP': '?' }, { 'POS': 'ADJ', 'OP': '*' }, { 'POS': 'NOUN', 'OP': '+' } ]
    # put the pattern into matcher
    matcher.add('light_verb', None, pattern)
    matcher.add('light_verb', None, pattern1)
    #put the sentences
    matches = matcher(nlp_doc)
    for match_id, start, end in matches:
        verb = nlp_doc[start]
        span = nlp_doc[start:end]
        if verb.text == 'do' or verb.text == 'get' or verb.text == 'give' or verb.text == 'have' or verb.text == 'make' or verb.text == 'take':
            with open(filename) as csvfile:
                readCSV = csv.reader(csvfile, delimiter=',')
                for row in readCSV:
                    if span.text == row[0]:
                        print(span.text)
                        string = string.replace(span.text, row[1])
            csvfile.close
    return string

```

Figure 3.20: code of extract light verb construction.

Before extract the light verb construction

joy	ok take a shower have be call name for about an hour think indulge respond now while allow robin thicke to serenade
joy	the plan be to take a nap yea still feel last night then go watch laker game smwhere fun look like no nap
joy	i have a feeling today be go to be amazing
sadness	can give a demo license expire yesterday
anger	for pete sake woman get a grip on

Figure 3.21: sentiment analysis dataset before extract the light verb construction

After extract the light verb construction

joy	ok shower have be call name for about an hour think indulge respond now while allow robin thicke to serenade
joy	the plan be to nap yea still feel last night then go watch laker game smwhere fun look like no nap
joy	i feel today be go to be amazing
sadness	can demonstrate license expire yesterday
anger	for pete sake woman grip on

Figure 3.22: sentiment analysis dataset after extract the light verb construction

3.3.3 Format Data

All the data in both datasets will be structured and convert the file format into csv file format.

Light Verb Construction Dataset

This dataset is using to extract the light verb construction from the sentiment analysis dataset, therefore it is no go to model to train and predict.

A	B
do a big deal	deal
do a bizarre dance	dance
do a brisk trade	trade
do a clothe switch	switch
do a community work	work
do a complete rehearsal	rehearse
do a comprehensive study	study

Figure 3.23: light verb construction dataset in csv file format

Sentiment Analysis Dataset

The sentiment analysis dataset have two different dataset, which is before extract the light verb construction and after extract the light verb construction. This is because need to perform the comparison between them.

	emotion	sentence
0	anger	get a bone to pick with one club
1	anger	get off back zimberalda
2	anger	hate rant and rave never like thing do
3	anger	alain be in a black mood and marguerite be try...
4	anger	the frustration and worry feel at the moment b...

Figure 3.24: sentiment analysis dataset in csv file format

3.4 Modeling

The modeling is important and core of this project, it is responsible to satisfy the project goals. The tasks in this phase is consist of select modeling technique, generate test design, and build the model.

3.4.1 Select Modeling Technique

Although there many suitable model to use for perform the sentiment analysis, only the three algorithm will be discuss in this paper.

- Linear Support Vector Machine (SVM)
- Multinomial Naïve Bayes
- Logistic Regression

3.4.2 Generate Test design

The sentiment dataset will be split into two data, which are training data, and testing data. The testing data will be using model to predict in order to get the accuracy. At the same, this project also determine and compare the performance of different ratio of training data and testing data. Therefore, the ratio are 70:30 and 80:20, as shown in table 3.4. And the coding as shown in figure 3.25.

Data	Ration
X_train1, X_test1, y_train1, y_test1	80:20
X_train, X_test, y_train, y_test	70:30

Table 3.4: Ratio of training data and testing data.

```
X_train1, X_test1, y_train1, y_test1 = train_test_split(df['sentence'], df['emotion'], test_size=0.2, random_state = 0)
X_train, X_test, y_train, y_test = train_test_split(df['sentence'], df['emotion'], test_size=0.3, random_state = 0)
```

Figure 3.25: coding of split the data into training data and testing data.

3.4.3 Build Model

The training data and testing data before fit into the model, the data require special preparation before used it for model. The TF-IDF is one of bag of word technique to help transform the text data to word frequency that can be used as into to estimator. The TF-IDF are try to highlight words that more interesting. The TF-IDF is use the TfidfVectorizer to token the text and encoding the text data. The figure 3.26 is shows the code of TF-IDF in two different data size. The N-gram range have help to compound two word, for example, the n-gram can handle 'not happy' to help to predict sadness emotion, if without n-gram, the 'not happy' become 'not', 'happy' then it will predict joy emotion.

```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer(sublinear_tf=True, min_df=5,max_df = 0.8, norm='l2', ngram_range=(1, 2))
X_features = tfidf.fit_transform(X_train).toarray()

tfidf1 = TfidfVectorizer(sublinear_tf=True, min_df=5,max_df = 0.8, norm='l2', ngram_range=(1, 2))
X_features1 = tfidf1.fit_transform(X_train1).toarray()
```

Figure 3.26: code of TF-IDF.

Linear Support Vector Machine (SVM)

The Linear SVM is using the from sklearn.svm import LinearSVC library the carry out the algorithms as shown in figure 3.27.

```
from sklearn.svm import LinearSVC

svmclf = LinearSVC(C=0.25).fit(X_features, y_train) # 70:30
svmclf1 = LinearSVC(C=0.25).fit(X_features1, y_train1) # 80:20
```

Figure 3.27: code of use the linear SVM algorithm.

Multinomial Naive Bayes

The Multinomial Naïve Bayes is using the from sklearn.naive_bayes import MultinomialNB library the carry the algorithms as shown in figure 3.28.

```
from sklearn.naive_bayes import MultinomialNB

NBclf = MultinomialNB().fit(X_features, y_train) # 70:30
NBclf1 =MultinomialNB().fit(X_features1, y_train1) # 80:20
```

Figure 3.28: code of use the Multinomial Naïve Bayes algorithm.

Logistic Regression

The Logistic Regression is using from sklearn.linear_model import LogisticRegression library the carry the algorithms as shown in figure 3.29.

```
from sklearn.linear_model import LogisticRegression  
  
lrclf = LogisticRegression(random_state=0).fit(X_features, y_train) # 70:30  
lrclf1 = LogisticRegression(random_state=0).fit(X_features1, y_train1)# 80:20
```

Figure 3.29: code of use the Logistic Regression algorithm

3.4.4 Assess Model

After fit data into the algorithms, the algorithms can predict the testing data, in order get the classification report of the algorithm, such as accuracy, precision, recall, and f1 score.

- Precision: precision is the ability of the model to not label the positive instances that are actually negative. For each class, it is defined as the ratio of the sum of true positives and true false positives. Therefore, it is possible to know the correct rate of all instances classified as positive.
- Recall: recall is the ability of the model to find all positive examples. For each class, it is defined as the ratio of true positives to the sum of true positives and false negatives. Therefore, it is possible to know the correct rate of all instances as actual positive.
- F1 score: F1 score is the weighted harmonic average of precision and recall. In general, the F1 score incorporates precision and recall into its computation, so it is lower than the accuracy measure.

In addition, it also show the confusion matrix visualization. The confusion matrix help to describe the performance of the model, such as accuracy, recall, error rate, and precision. Below will show two different dataset and different data size ratio.

Before Extract the Light Verb Construction

Linear Support Vector Machine (SVM):

The Linear SVM get accuracy after predict the testing data. In data size ratio 70:30, the linear SVM get 62.79% accuracy and in data size ratio 80:20, it get 63.55% as shown in figure 3.30. Compare this two different data size ratio, the accuracy of data size ratio 80:20 is higher than the data size ratio 70:30.

```

y_pred = svmclf.predict(tfidf.transform(X_test))
print("Linear SVM Accuracy Score -> {:.2f}".format(accuracy_score(y_pred, y_test)*100))

print('')
print('Ratio: 80:20')
y_pred1 = svmclf1.predict(tfidf1.transform(X_test1))
print("Linear SVM Accuracy Score -> {:.2f}".format(accuracy_score(y_pred1, y_test1)*100))

```

Ratio: 70:30
Linear SVM Accuracy Score -> 62.79

Ratio: 80:20
Linear SVM Accuracy Score -> 63.55

Figure 3.30: accuracy of two data size ratio in Linear SVM.

Moreover, the Linear SVM also show other performance and visualize the performance of classification report as shown in figure 3.31-3.32 Based on the result, the performance of data size ratio 80:20 is higher than the data size ratio 70:30.

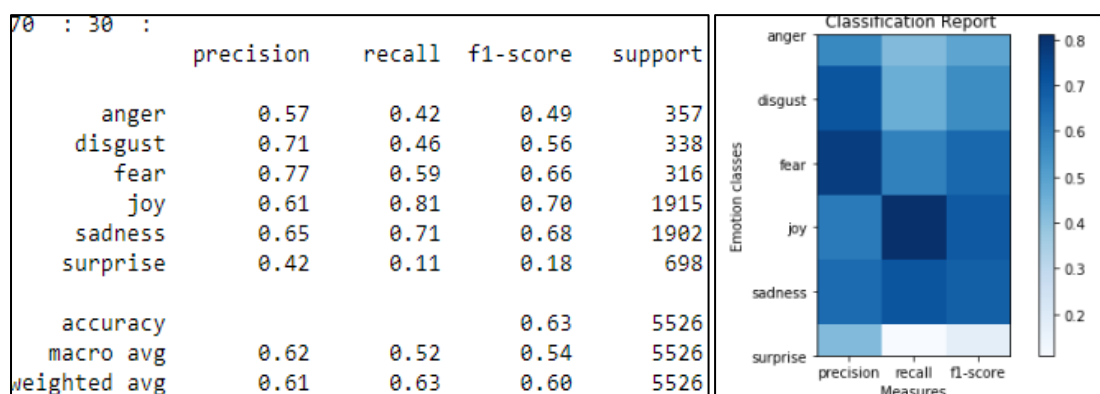


Figure 3.31: classification report of data size ratio 70:30 in Linear SVM.

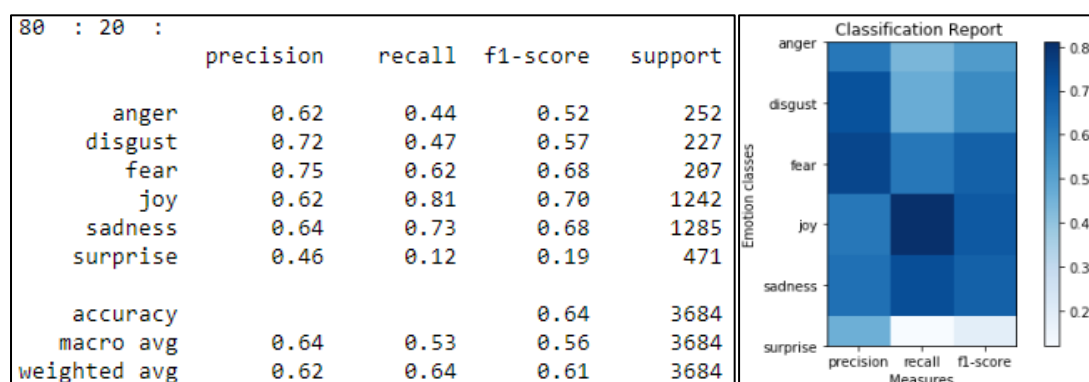


Figure 3.32: classification report of data size ratio 80:20 in Linear SVM.

Confusion Matrix

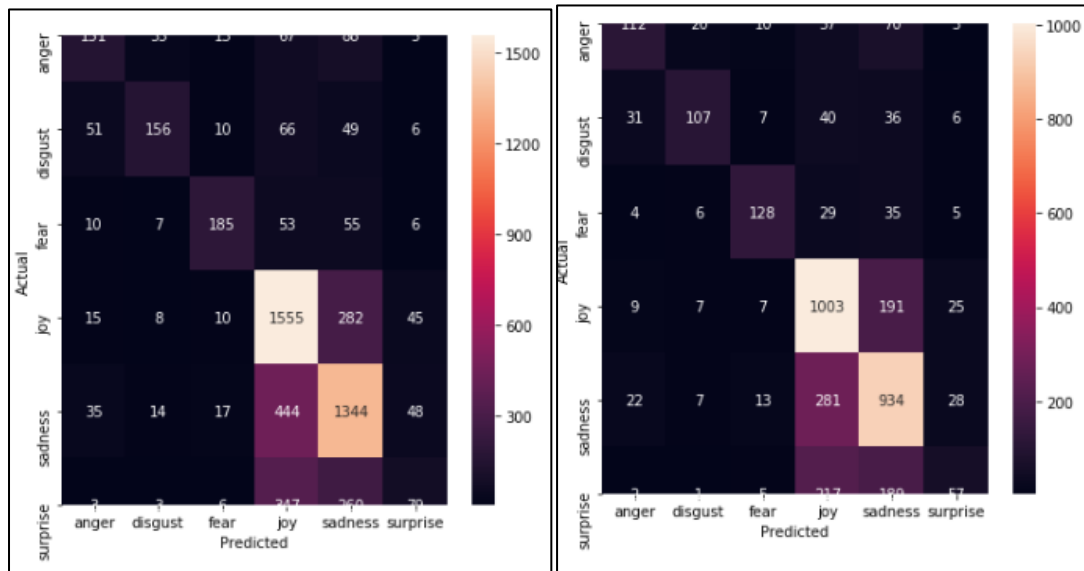


Figure 3.33: confusion matrix of two data size ratio (Left:70:30. Right: 80:20).

Multinomial Naive Bayes

The Multinomial Naïve Bayes get accuracy after predict the testing data. In data size ratio 70:30, the Multinomial Naïve Bayes get 60.26% accuracy and in data size ratio 80:20, it get 60.18% as shown in figure 3.34. Compare this two different data size ratio, the accuracy of data size ratio 80:20 is lower than the data size ratio 70:30.

```
print('Ratio: 70:30')
y_pred = NBclf.predict(tfidf.transform(X_test))
print("Multinomial NB Accuracy Score -> {:.2f}".format(accuracy_score(y_pred, y_test)*100))
print('Ratio: 80:20')
y_pred1 = NBclf1.predict(tfidf1.transform(X_test1))
print("Multinomial NB Accuracy Score -> {:.2f}".format(accuracy_score(y_pred1, y_test1)*100))
```

Ratio: 70:30
Multinomial NB Accuracy Score -> 60.26
Ratio: 80:20
Multinomial NB Accuracy Score -> 60.18

Figure 3.34: accuracy of two data size ratio in Multinomial Naïve Bayes.

Moreover, the Multinomial Naïve Bayes also show other performance and visualize the performance of classification report as shown in figure 3.35 -3.36. Based on the result, the performance of data size ratio 80:20 is close to the data size ratio 70:30.

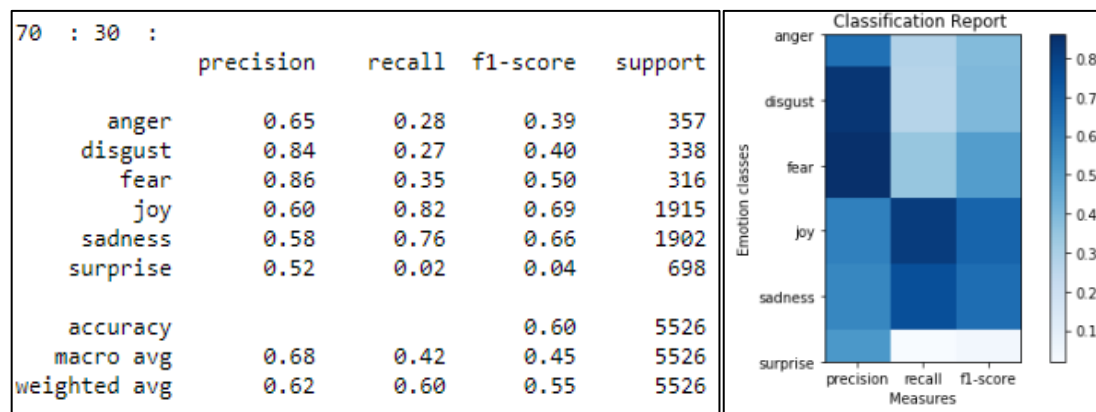


Figure 3.35: classification report of data size ratio 70:30 in Multinomial Naïve Bayes.

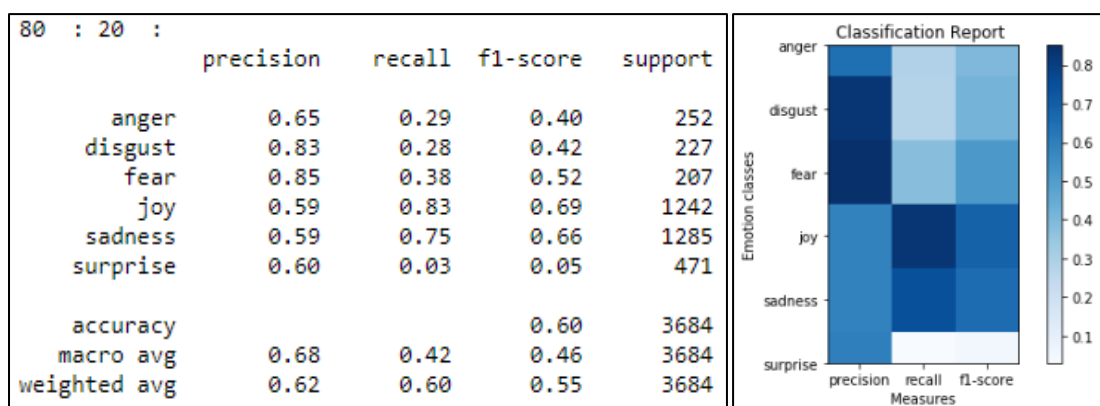


Figure 3.36: classification report of data size ratio 80:20 in Multinomial Naïve Bayes.

Confusion Matrix

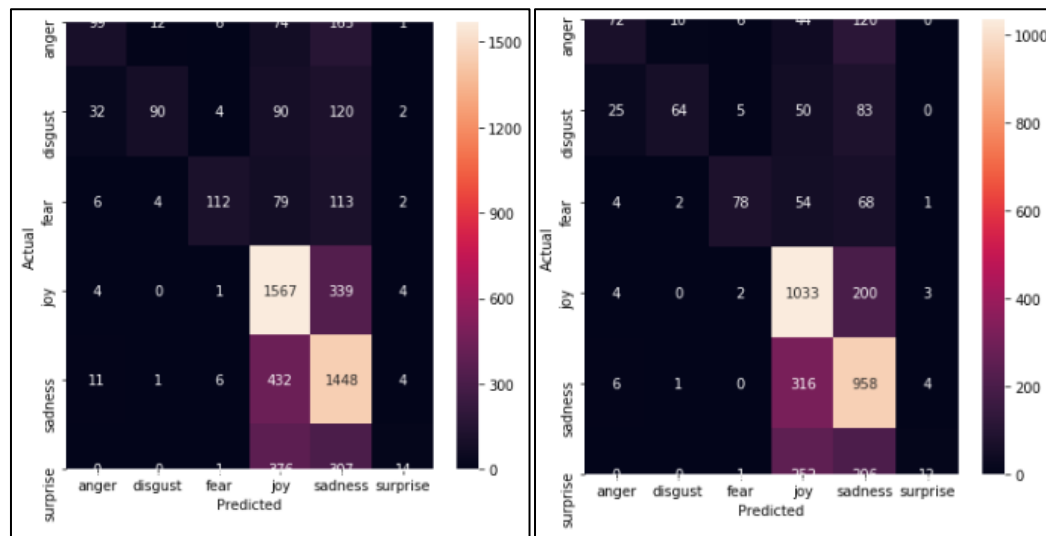


Figure 3.37: confusion matrix of different data size ratio (Left:70:30. Right: 80:20).

Logistic Regression

The Logistic Regression get accuracy after predict the testing data. In data size ratio 70:30, the Logistic Regression get 62.81% accuracy and in data size ratio 80:20, it get 62.98% as shown in figure 3.38. Compare this two different data size ratio, the accuracy of data size ratio 80:20 is higher than the data size ratio 70:30.

```
print('Ratio: 70:30')
y_pred = lrclf.predict(tfidf.transform(X_test))
print("LogisticRegression Accuracy Score -> {:.2f}".format(accuracy_score(y_pred, y_test)*100))

print('')
print('Ratio: 80:20')
y_pred1 = lrclf1.predict(tfidf1.transform(X_test1))
print("LogisticRegression Accuracy Score -> {:.2f}".format(accuracy_score(y_pred1, y_test1)*100))

Ratio: 70:30
LogisticRegression Accuracy Score -> 62.81

Ratio: 80:20
LogisticRegression Accuracy Score -> 62.98
```

Figure 3.38: accuracy of two different data size ratio in Logistic Regression.

Moreover, the Logistic Regression also show other performance and visualize the performance of classification report as shown in figure 3.39-3.40. Based on the result, the performance of data size ratio 80:20 is higher than the data size ratio 70:30.

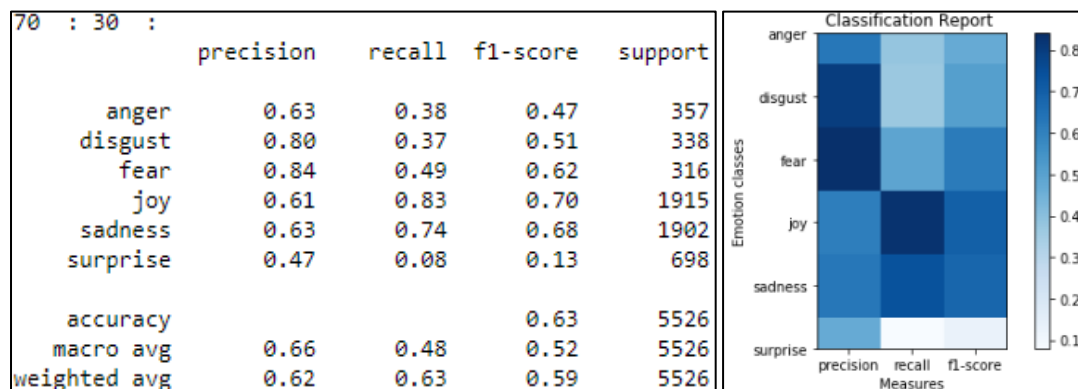


Figure 3.39: classification report of data size ratio 70:30 in Logistic Regression.

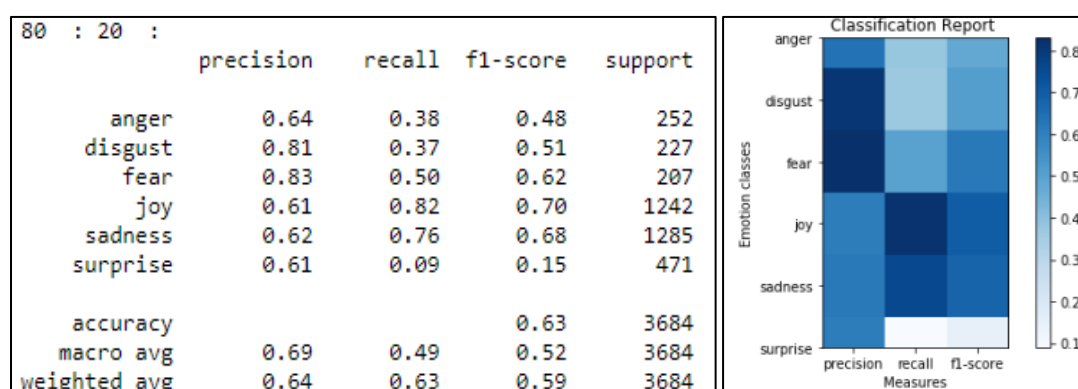


Figure 3.40: classification report of data size ratio 80:20 in Logistic Regression.

Confusion Matrix

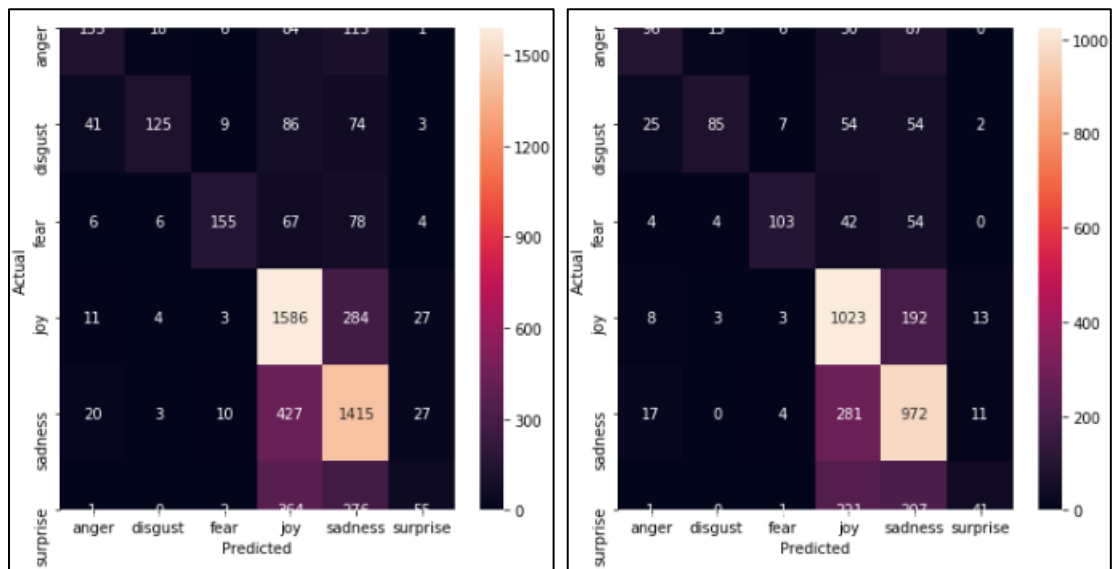


Figure 3.41: confusion matrix of two data size ratio (Left:70:30. Right: 80:20).

After Extract the Light Verb Construction

The next dataset is extract the light verb construction, the data size also will be distinguish two different in order to compare the performance.

Linear Support Vector Machine (SVM)

The Linear SVM get accuracy after predict the testing data. In data size ratio 70:30, the linear SVM get 65.74% accuracy and in data size ratio 80:20, it get 66.23% as shown in figure 3.42. Compare this two different data size ratio, the accuracy of data size ratio 80:20 is higher than the data size ratio 70:30.

```
print('Ratio: 70:30')
y_pred = svmclf.predict(tfidf.transform(X_test))
print("Linear SVM Accuracy Score -> {:.2f}".format(accuracy_score(y_pred, y_test)*100))
print('')
print('Ratio: 80:20')
y_pred1 = svmclf1.predict(tfidf1.transform(X_test1))
print("Linear SVM Accuracy Score -> {:.2f}".format(accuracy_score(y_pred1, y_test1)*100))
```

Ratio: 70:30
Linear SVM Accuracy Score -> 65.74

Ratio: 80:20
Linear SVM Accuracy Score -> 66.23

Figure 3.42: accuracy of two different data size ratio in Linear SVM.

Moreover, the Linear SVM also show other performance and visualize the performance of classification report as shown in figure 3.43-3.44. Based on the result, the performance of data size ratio 80:20 is higher than the data size ratio 70:30.

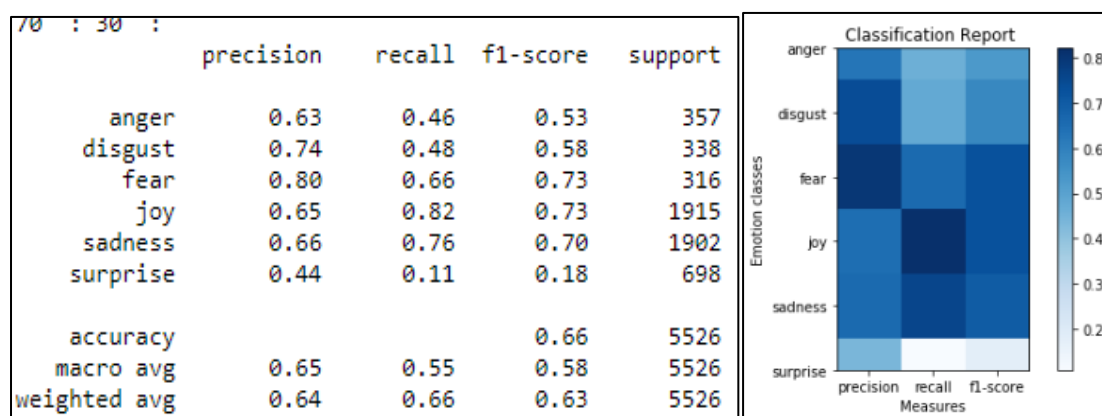


Figure 3.43: classification report of data size ratio 70:30 in Linear SVM.

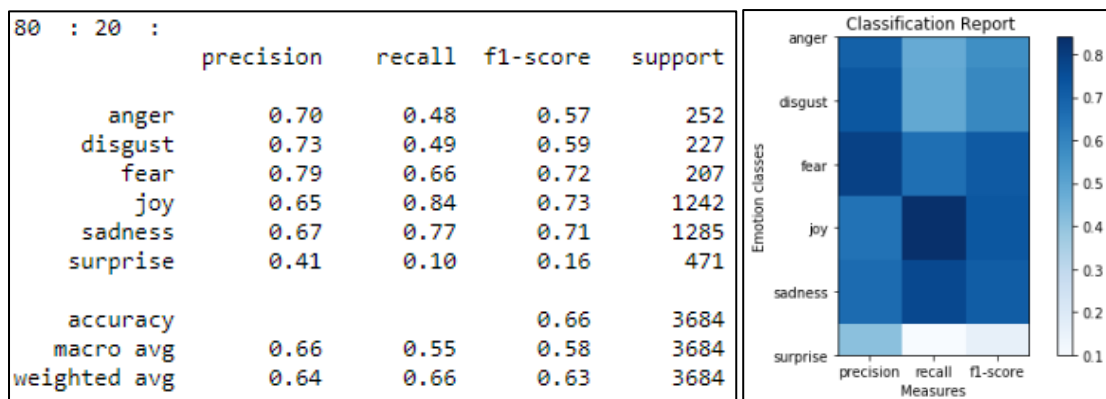


Figure 3.44: classification report of data size ratio 80:20 in Linear SVM.

Confusion Matrix

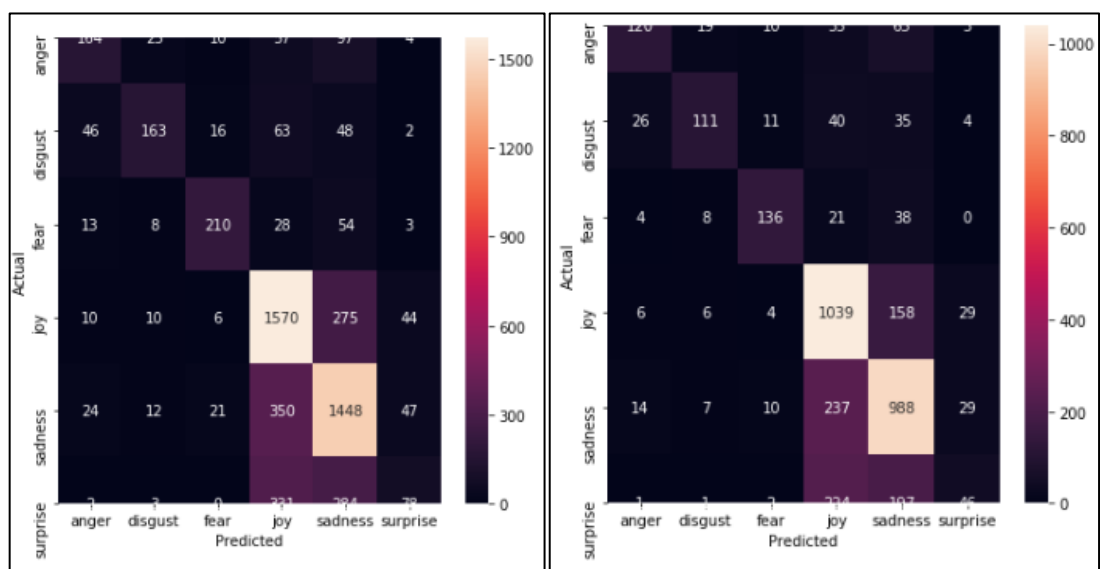


Figure 3.45: confusion matrix of two data size ratio (Left:70:30. Right: 80:20).

Multinomial Naive Bayes

The Multinomial Naïve Bayes get accuracy after predict the testing data. In data size ratio 70:30, the Multinomial Naïve Bayes get 60.48% accuracy and in data size ratio 80:20, it get 60.78% as shown in figure 3.46. Compare this two different data size ratio, the accuracy of data size ratio 80:20 is higher than the data size ratio 70:30.

```
print('Ratio: 70:30')
y_pred = NBclf.predict(tfidf.transform(X_test))
print("Multinomial NB Accuracy Score -> {:.2f}".format(accuracy_score(y_pred, y_test)*100))
print('Ratio: 80:20')
y_pred1 = NBclf1.predict(tfidf1.transform(X_test1))
print("Multinomial NB Accuracy Score -> {:.2f}".format(accuracy_score(y_pred1, y_test1)*100))
```

Ratio: 70:30
Multinomial NB Accuracy Score -> 60.48

Ratio: 80:20
Multinomial NB Accuracy Score -> 60.78

Figure 3.46: accuracy of two different data size ratio in Multinomial Naïve Bayes.

Moreover, the Multinomial Naïve Bayes also show other performance and visualize the performance of classification report as shown in figure 3.47-3.48. Based on the result, the performance of data size ratio 80:20 is close to the data size ratio 70:30.

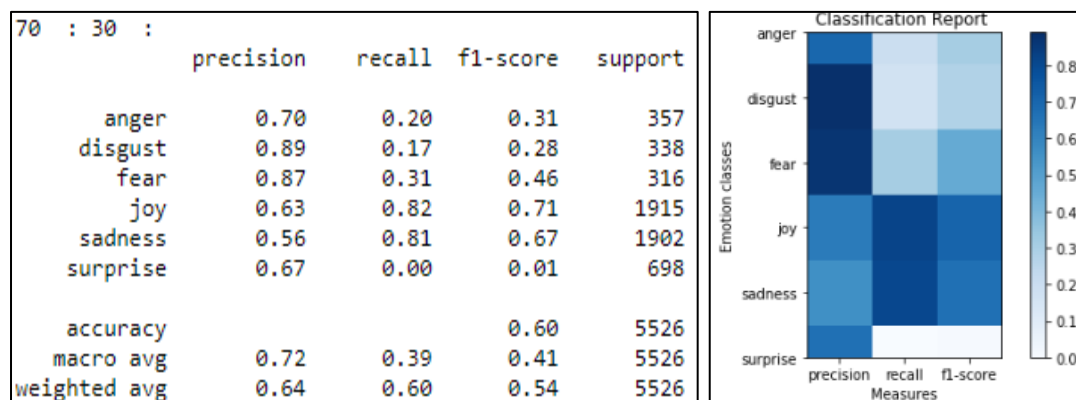


Figure 3.47: classification report of data size ratio 70:30 in Multinomial Naïve Bayes.

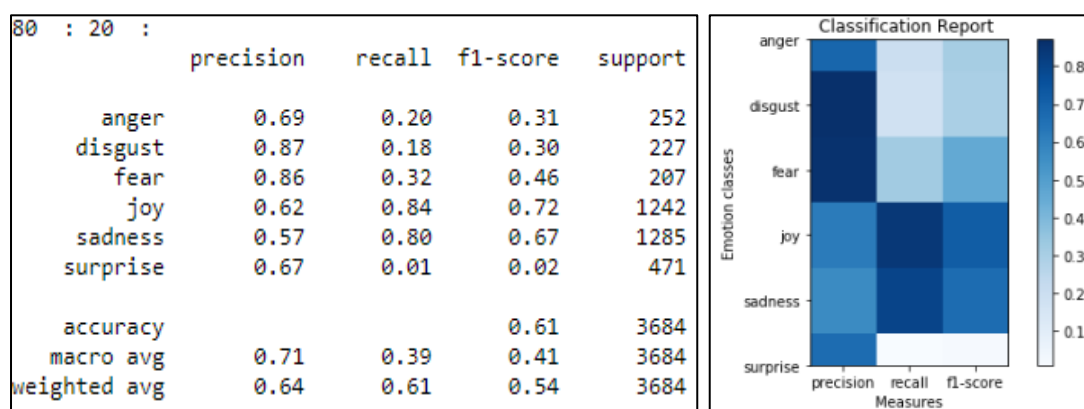


Figure 3.48: classification report of data size ratio 80:20 in Multinomial Naïve Bayes.

Confusion Matrix

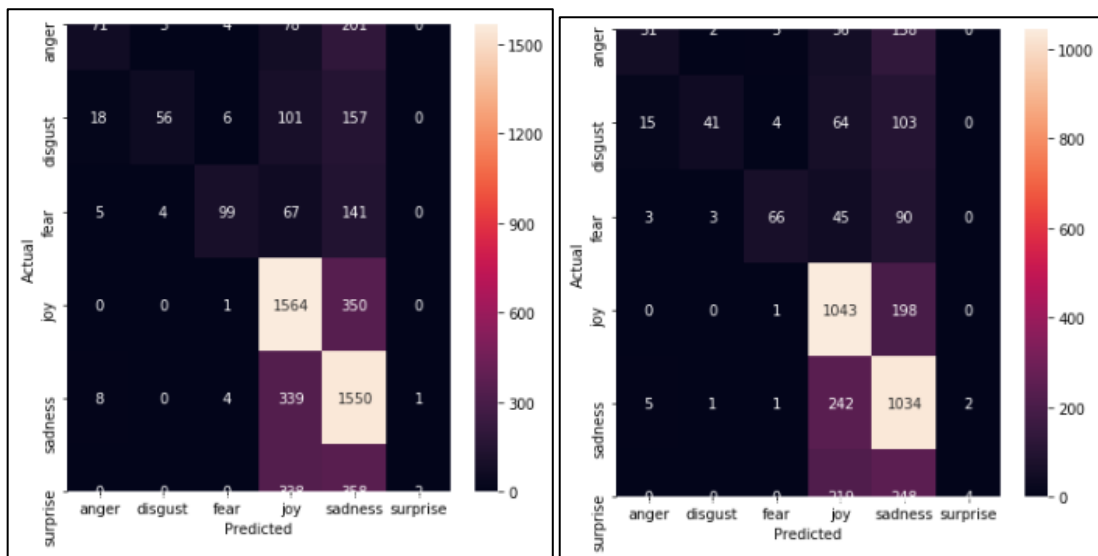


Figure 3.49: confusion matrix of two data size ratio (Left:70:30. Right: 80:20).

Logistic Regression

The Logistic Regression get accuracy after predict the testing data. In data size ratio 70:30, the Logistic Regression get 64.39% accuracy and in data size ratio 80:20, it get 65.15% as shown in figure 3.50. Compare this two different data size ratio, the accuracy of data size ratio 80:20 is higher than the data size ratio 70:30.

```
print('Ratio: 70:30')
y_pred = lrclf.predict(tfidf.transform(X_test))
print("LogisticRegression Accuracy Score -> {:.2f}".format(accuracy_score(y_pred, y_test)*100))

print('Ratio: 80:20')
y_pred1 = lrclf1.predict(tfidf1.transform(X_test1))
print("LogisticRegression Accuracy Score -> {:.2f}".format(accuracy_score(y_pred1, y_test1)*100))
```

Ratio: 70:30
LogisticRegression Accuracy Score -> 64.39

Ratio: 80:20
LogisticRegression Accuracy Score -> 65.15

Figure 3.50: accuracy of two different data size ratio in Logistic Regression.

Moreover, the Logistic Regression also show other performance and visualize the performance of classification report as shown in figure 3.51-3.52. Based on the result, the performance of data size ratio 80:20 is higher than the data size ratio 70:30.

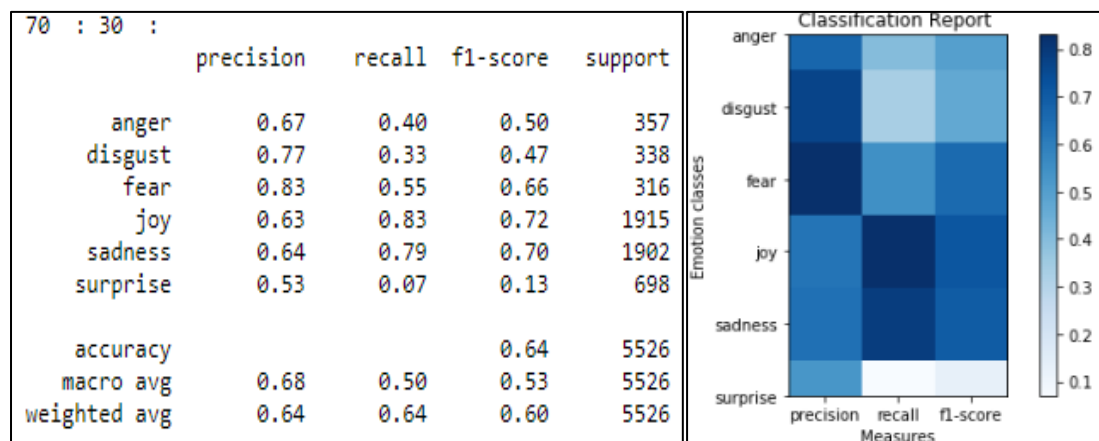


Figure 3.51: classification report of data size ratio 70:30 in Logistic Regression.

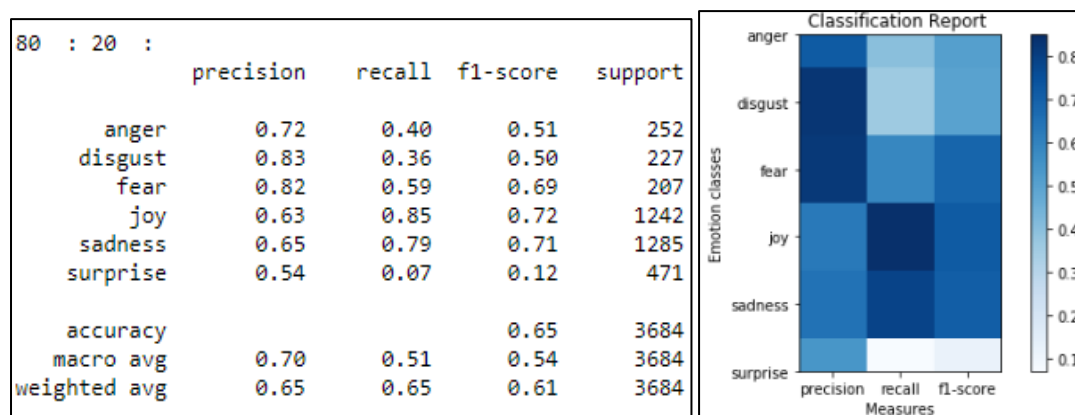


Figure 3.52: classification report of data size ratio 80:20 in Logistic Regression.

Confusion Matrix

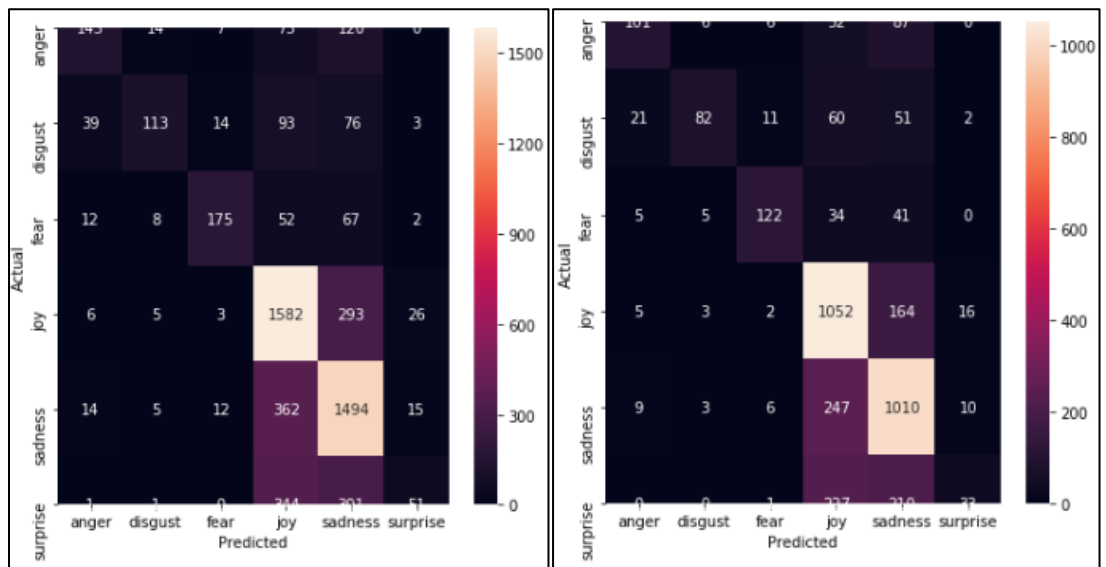


Figure 3.53: confusion matrix of two data size ratio (Left:70:30. Right: 80:20).

Before extract the light verb construction

The figure 3.54 shows the performance of three model with using dataset without extract the light verb construction. Based on the figure, the Linear SVM have provide high performance compared to other two model. The multinomial Naïve Bayes have a poor performance in this project. The performance of both tree model of data size ratio 80:20 was higher than the data size ratio 70:30.

Model	70 : 30					80 : 20				
	Accuracy(%)	Classes	Precision	Recall	F1 score	Accuracy(%)	Classes	Precision	Recall	F1 score
Linear SVM	62.79	anger	0.57	0.42	0.49	63.55	anger	0.62	0.44	0.52
		disgust	0.71	0.46	0.56		disgust	0.72	0.47	0.57
		fear	0.77	0.59	0.66		fear	0.75	0.62	0.68
		joy	0.61	0.81	0.70		joy	0.62	0.81	0.70
		sadness	0.65	0.71	0.68		sadness	0.64	0.73	0.68
		surprise	0.42	0.11	0.18		surprise	0.46	0.12	0.19
Logistic Regression	62.81	anger	0.65	0.28	0.39	62.98	anger	0.65	0.29	0.40
		disgust	0.84	0.27	0.40		disgust	0.83	0.28	0.42
		fear	0.86	0.35	0.50		fear	0.85	0.38	0.52
		joy	0.60	0.82	0.69		joy	0.59	0.83	0.69
		sadness	0.58	0.76	0.66		sadness	0.59	0.75	0.66
		surprise	0.52	0.02	0.04		surprise	0.60	0.03	0.05
Multinomial NB	60.26	anger	0.65	0.28	0.39	60.18	anger	0.65	0.29	0.40
		disgust	0.84	0.27	0.40		disgust	0.83	0.28	0.42
		fear	0.86	0.35	0.50		fear	0.85	0.38	0.52
		joy	0.60	0.82	0.69		joy	0.59	0.83	0.79
		sadness	0.58	0.76	0.66		sadness	0.59	0.75	0.66
		surprise	0.52	0.02	0.04		surprise	0.60	0.03	0.05

Figure 3.54: performance of three model in dataset before extract light verb construction.

After extract the light verb construction

The figure 3.55 shows the performance of three model with using dataset with extract the light verb construction. Based on the figure, the Linear SVM have provide high performance compared to other two model. The multinomial Naïve Bayes have a poor performance in this project. The performance of both three model of data size ratio 80:20 was higher than the data size ratio 70:30.

Model	70 : 30					80 :20				
	Accuracy(%)	Classes	Precision	Recall	F1 score	Accuracy(%)	Classes	Precision	Recall	F1 score
Linear SVM	65.74	anger	0.63	0.46	0.53	66.23	anger	0.70	0.48	0.57
		disgust	0.74	0.48	0.58		disgust	0.73	0.49	0.59
		fear	0.80	0.66	0.73		fear	0.79	0.66	0.72
		joy	0.65	0.82	0.73		joy	0.65	0.84	0.73
		sadness	0.66	0.76	0.70		sadness	0.67	0.77	0.71
		surprise	0.44	0.11	0.18		surprise	0.41	0.10	0.16
Logistic Regression	64.39	anger	0.67	0.40	0.50	65.15	anger	0.72	0.40	0.51
		disgust	0.77	0.33	0.47		disgust	0.83	0.36	0.50
		fear	0.83	0.55	0.66		fear	0.82	0.59	0.69
		joy	0.63	0.83	0.72		joy	0.63	0.85	0.72
		sadness	0.64	0.79	0.70		sadness	0.65	0.79	0.71
		surprise	0.53	0.07	0.13		surprise	0.54	0.07	0.12
Multinomial NB	60.48	anger	0.70	0.20	0.31	60.78	anger	0.69	0.20	0.31
		disgust	0.89	0.17	0.28		disgust	0.87	0.18	0.30
		fear	0.87	0.31	0.46		fear	0.86	0.31	0.46
		joy	0.63	0.82	0.71		joy	0.62	0.84	0.72
		sadness	0.56	0.81	0.67		sadness	0.57	0.80	0.67
		surprise	0.67	0.00	0.01		surprise	0.67	0.01	0.02

Figure 3.55: performance of three model in dataset After extract light verb construction.

Summarize the result

After get performance from three model, which are Linear SVM, Multinomial Naïve Bayes, and Logistic Regression. The two figure above was summarize the result and ranking the models based on the performance. In this two model, the Linear SVM have provide high accuracy. Moreover, the recall and f1 score for the class 'surprise' have a low score on each of models, this may because the sentences of class 'surprise' was similar to the class 'joy', it make the model hard to classify, thereby affect the accuracy. This mean that, it also affect the performance of class 'joy'.

3.5 Evaluation

The evaluation phase is to determine whether there are some important issues that have not been sufficiently considered. The tasks consist of evaluate result, review process.

3.5.1 Evaluate Result

This project goal is to improve the accuracy of sentiment analysis by detecting the light verb construction, to determine the light verb construction have contribute to the sentiment analysis. The Figure 3.56 shows that the improvement of the extract light verb construction. Based on the figure, the Linear SVM, Logistic have improve the accuracy more than 1%. However, the improvement of Naive Bayes was lower than the other model. This mean that it is achieve the project goal and show the light verb construction have impact on the sentiment analysis. The extract the light verb construction from the sentence to simply the sentence, to make the model easier to target the important word thereby improve the accuracy.

	70 : 30			80 :20		
	Before extract LVC	After extract LVC		Before extract LVC	After extract LVC	
Model	Accuracy (%)	Accuracy (%)	Improvement (%)	Accuracy (%)	Accuracy (%)	Improvement(%)
Linear SVM	62.79	65.74	2.95	63.55	66.23	2.68
Logistic Regression	62.81	64.39	1.58	62.98	65.15	2.17
Multinomial NB	60.26	60.48	0.22	60.18	60.78	0.60

Figure 3.56: improvement of accuracy of model

3.5.2 Review Process

After evaluating the result, the result of models was satisfactory and to satisfy the project goal. Then, go to process there review to check any important factor have overlooked. Based on the result, the precision and recall of the class 'surprise' sentence is too close to class 'joy', this means that the sentences of class 'surprise' need to increase new sample in order the model may improve. In addition, the sample of light verb construction also less, therefore require to increase the number of light verb construction in the dataset. Moreover, the light verb construction in the dataset is consist of the adjective and adverb. After extracting the light verb construction from the sentence and replace to the same meaning single verb word. This may lose some important word in the sentiment analysis. This is because some adjective may contribute to the sentiment analysis.

3.6 Deployment

In general, the generation of models is not end of the project, the deployment phase is plan to develop a live model to perform the sentiment analysis with detecting the light verb construction. The task in this phase is plan deployment.

3.6.1 Plan Deployment

After train the models, the project is plan to develop a web-based system to allow the user to upload the sentences to let the model to predict and classify the sentiment and generate a result to the user. In this process, the user choose the one model to perform the classification. The sentence upload by the user will the process text preprocessing which include the remove HTML tag, URL link, punctuations, tag username, accents, replace the emoji to word and lower case the word. Not only that, the sentences also perform lemmatization in order to extracting the light verb construction thereby to improve the accuracy and satisfy the user.

3.7 Functional Requirement

1 Sentiment analysis module

- 1.1 The system should be able to allow user to select the classification model
- 1.2 The system should be able to allow user to upload the data to the system.
- 1.3 The system should be able to text preprocessing the data.
- 1.4 The system should be able to perform the sentiment analysis by classification model.
- 1.5 The system should be able to display the result to the user.
- 1.6 The system should be able to display the bar chart to the user.

2 Light verb construction module

- 2.1 The system should be able to perform the lemmatization to the data.
- 2.2 The system should be able to detect the light verb construction in the sentences.
- 2.3 The system should be able to extract the light verb construction from the sentences.
- 2.4 The system should be able to replace the light verb construction to a single same meaning word.

3.8 Non-Functional Requirement

- 1 Usability: The user interface of system will design as simple and clear, to allow user able to learn and understand the system.
- 2 Availability: The system able to allow user to perform all the function and system can be operate 24/7.
- 3 Effectiveness: The system able to allow user to use it to achieve their goal, to get the result of the sentiment analysis.
- 4 Performance: The system will be provide the result to the user within 2 minutes.

3.9 Chapter Summary and Evaluation

In this chapter, understand about the CRISP-DM methodology to help to this project to achieve the project objective and goals, which is improve the accuracy the sentiment analysis by detecting the light verb construction and the performance of three model also prove that the light verb construction extraction have contributed to sentiment analysis. In the process of the methodology also help to understand the light verb construction work to the sentences and affect of the sentiment analysis.

In addition, in the process of methodology, there are many problems. The first is the light verb structure dataset. This is because the light verb structure is not predictive and difficult to find on the website. Based on various literature reviews, the more frequently used dataset in their project is the dataset created by Tu and Roth in the British National Corpus (BNC). However, this is the dataset they used to perform classification, which will be distinguished from this project. Therefore, more time is needed to understand this dataset to perform clean data in order to perform detection and extract light verb constructions from sentences.

The second problem is to understand the pattern of light verb construction, this is because the light verb construction is changeable. There are many constructions was similar to light verb construction but no the light verb, such as 'take a bag' and 'take a while' in addition, after do some research, the light verb construction also need to consider about the adjective, for example 'take a short break' and 'make a nice decision', thereby to detect more light verb construction from the sentences.

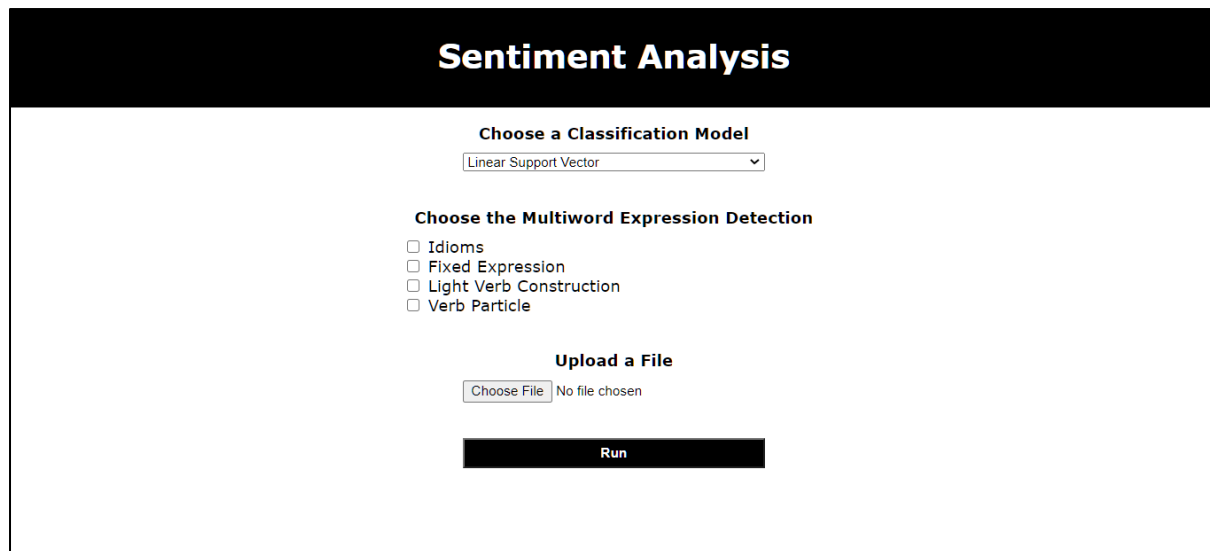
Chapter 4

System Design

4. System Design

This chapter will describe some design of the project, which are the User Interface diagram, system architectural diagram, and activity diagram,

4.1 User Interface Diagram



The screenshot shows a web interface titled "Sentiment Analysis". It features a dropdown menu for "Choose a Classification Model" with "Linear Support Vector" selected. Below this is a section for "Choose the Multiword Expression Detection" with four unchecked checkboxes: "Idioms", "Fixed Expression", "Light Verb Construction", and "Verb Particle". Further down is an "Upload a File" section with a "Choose File" button and the text "No file chosen". At the bottom is a large black button labeled "Run".

Figure 4.1: Main page of the sentiment analysis

The Figure 4.1 is showing the main page of the sentiment analysis page. The user interface is created in the flask with the HTML language. The user interface page design as simple and clear, user can be clearly to use the system to perform the sentiment analysis. The top of page is show the system name. Then, the model is using the dropdown list to user to choose the model, this can avoid the user make the mistake and reduce the error occur. Besides that, the user can be allow tick the checkbox to perform the Multiword Expression detection. Next, the user also can allow to choose the text file upload to the system, and click the run to process the sentiment analysis. The structure design of the content is from top to down, to make the user feel satisfy.

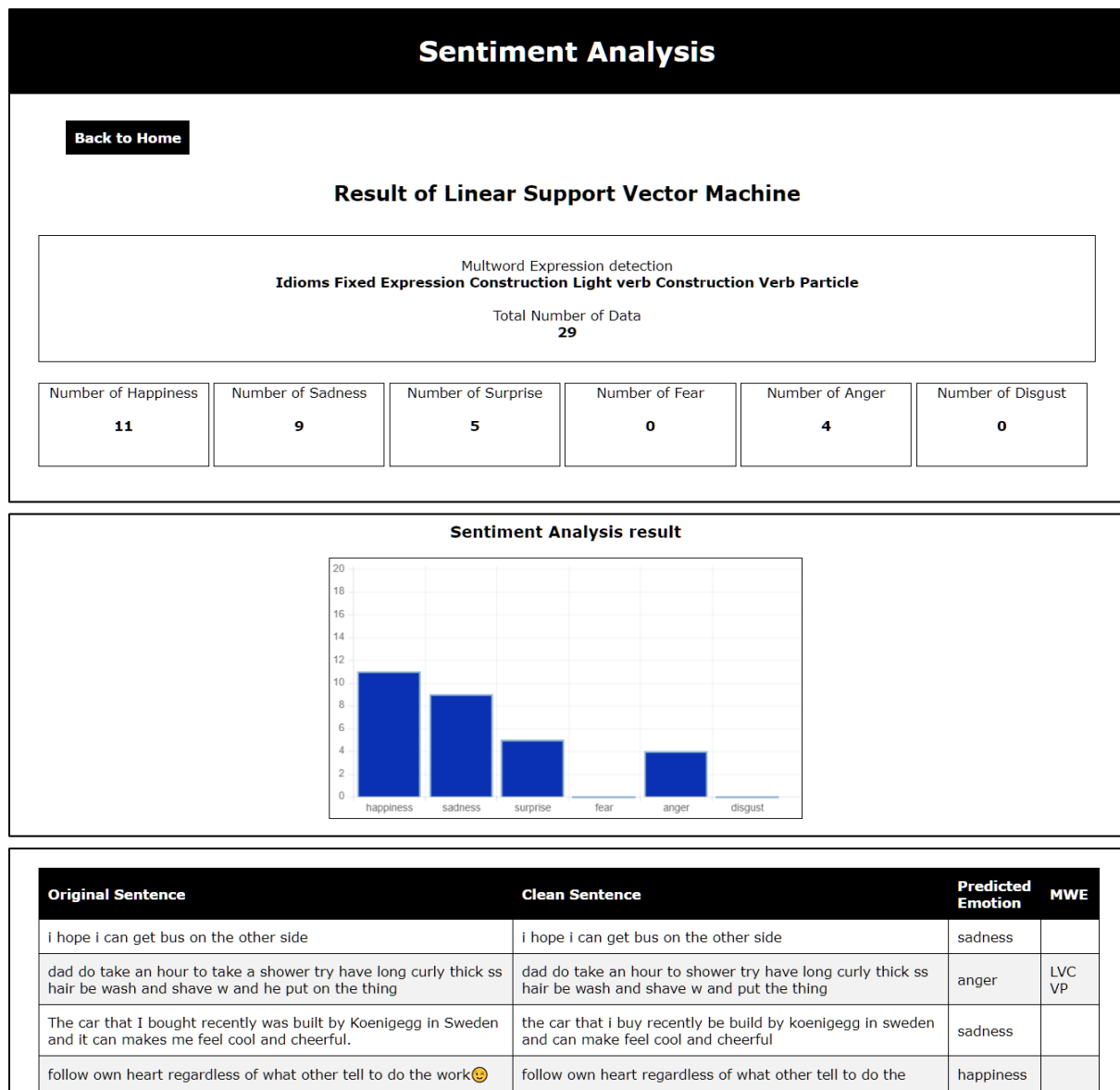


Figure 4.2: Result page of the sentiment analysis

The Figure 4.2 is showing the result page of the sentiment analysis page. The user interface is created in the flask with the HTML language. The user interface page is provide the result to users. The top of page is show the system name. The page will showing the classification model choose and the Multiword Expression choose. The page also will number of total data, and number of each six emotion in the result. Besides that, the page also will build a bar chart to make the user more easily to understand the result. Last, the page also provide the table that consist the original sentences, cleaned sentences and the result.

4.2 System Architectural Design

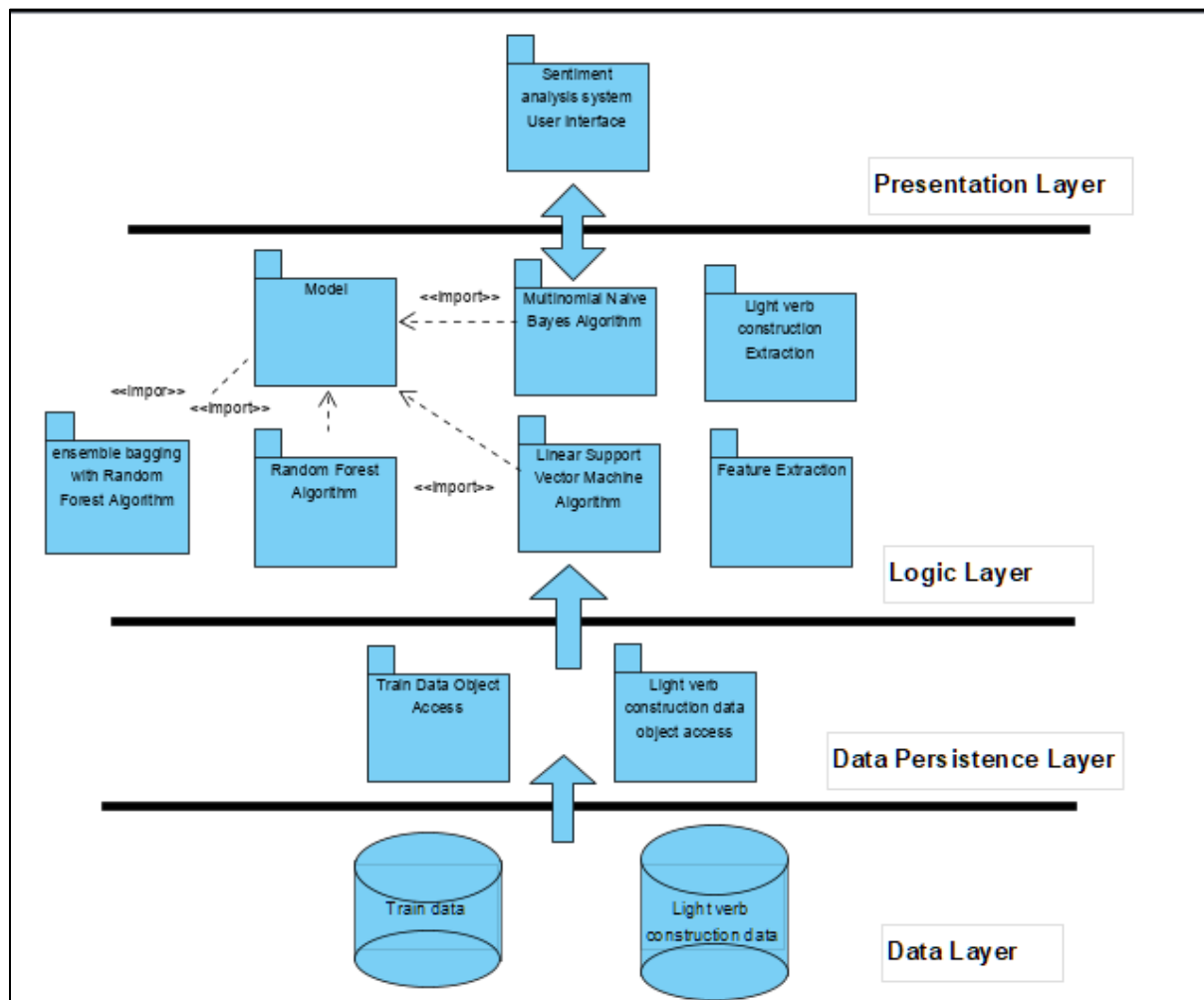


Figure 4.3: layered architecture diagram of the sentiment analysis

The figure 4.3 is showing the layered architecture diagram of the sentiment analysis. The diagram is constructed by 4 layers, which are presentation, logic, data persistence, data layer. This diagram can provide the abstraction for the view of the system as a whole.

The presentation layer is the layer which includes the sentiment analysis system user interface. It is display the information or the result to user and accepting the input from the user. Other than that, the presentation layer will communicate with the logic layer to pass the input data that is get from the presentation layer in order to perform the tasks. Logic layer is the layer which include the business rule, data validation and how the component interact with other component. Components in the logic layer are model, linear Support Vector Machine algorithm, Random Forest algorithm, Multinomial Naive Bayes algorithm ensemble bagging with Random Forest algorithm, light verb construction extraction, feature extraction. The four

algorithm have import relationship with the model, this is because based on the user choice, and the system will be perform sentiment analysis with selected model.

Data persistence layer is the layer which includes the database communication, extract data. The data persistence layer will be communicate with data layer in order to carry out the data persistence process. There are two data access objects in the layer, which are the train data access object, light verb construction object. The data access objects are responsible for data persistence of each of object data going out from the dataset. For example, the train data access object will be used to take the data from the dataset and fit into selected model, the light verb construction data access object will used to perform the light verb construction extraction. The data layer is the layer include the dataset of the system. This layer have two dataset, which is the train data and light verb construction data.

.

4.3 Activity Diagram

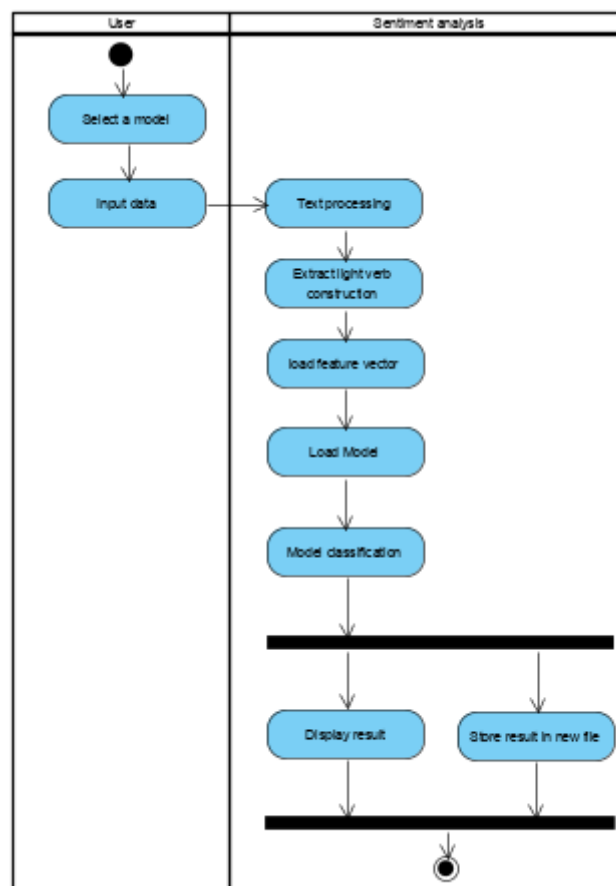


Figure 4.4 Activity diagram of sentiment analysis

The figure 4.4 is showing the activity diagram of the activity diagram of the sentiment analysis. The system is allow the select a model, which are the Linear Support Vector Machine, Random Forest, Multinomial Naïve Bayes and ensemble bagging with random forest algorithm. Then, the user can input the data, which is consist of sentences to the system. The data will be process the text processing, which is include the remove HTML tag, URL link, @username, accents, punctuation, replace emoji to word and convert to lowercase. Then will process the light verb construction extraction. After that, load a feature vector pickle file, which is the TfidfVectorizer pickle and extract input data. Then, the system will load selected model pickle file. Next, the input data will be process the classification in order to get the result. Last, the system will display the result and bar chart to the user and store the result in the new csv file.

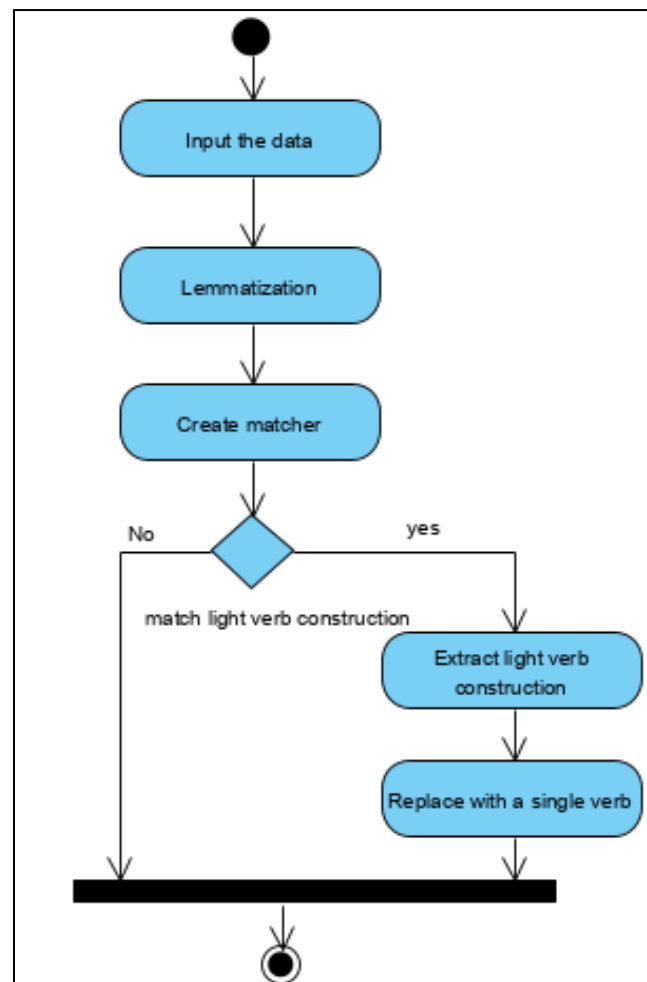


Figure 4.5 Activity diagram of light verb construction extraction

The Figure 4.5 is showing the activity diagram of the light verb construction extraction. After text processing, the data will pass into the light verb construction extraction. The system will use the spaCy library to process the lemmatization for each of the sentences in the data. The lemmatization to make sure each word in the sentence becomes a base word, such as 'makes', 'making', 'made' convert to the 'make'. This process is helpful when extracting the light verb construction from the sentences. The system also uses a rule-based matching approach to detect the light verb construction, which is to create a matcher and set the pattern into the matcher. The pattern is based on the semantic of the light verb construction and it is considered about the adjective and adverb. When the rule-based matching approach is matching the light verb construction in the sentences, it processes extraction and replacement. The light verb construction in the sentences will be extracted and replaced with a single same meaning word, such as 'make a decision' replace with 'decide'.

4.4 Chapter Summary and Evaluation

In this chapter, it is describe the user interface diagram, layer architecture diagram, activity diagram. The user interface will be created in the flask with HTML language. It is allow any browser to display the page to user. The architecture diagram and activity diagram were draw by using the visual paradigm. Found that there are many type of system architecture diagram, only the layered architecture diagram is selected and describe. This is because it can provide enough detail to understand the role and to understand what each of the components does and its relationship. The layer architecture diagram is easy to implement this is because it is no require any assumption to be model regarding the method, properties and other. The activity diagram is described to understand the flow of the system. Based on the activity diagram, it allow reader fast to understand because it is simple.

Chapter 5

Implementation and Testing

5. Implementation and Testing

This chapter will be discuss about the system implementation and the testing. The system will be implement in the web based system by using the flask. After the system implementation, the system will be testing in order to make sure the system is perform well.

5.1 Implementation

The implementation of the sentiment analysis system will be implement in order to fulfil the user needs and requirement.

Team Member	Multiword Expression	Best Classification Model
Liew Er Wei	Light Verb Construction	Linear Support Vector
Low Hong Ming	Idioms	Random Forest
Fung Wai Lun	Fixed Expression	Ensemble Bagging with Random Forest
Phang Jia Luo	Verb Practice	Naïve Bayes

Table 5.1: Task responsible table

Before start the implement the sentiment analysis system, the project team member was has different Multiword Expressions detection and the best classification model (refers to the table 5.1).

5.1.1 Pickle File

In chapter 3, the project has developed three classification model for the sentiment analysis. However, if the sentiment analysis system every time need the train the model during the users run the system. It requires a long time taken. This will make the users feel trouble and unsatisfied. Therefore, in order to save the time taken and improve the response speed, the pickle file can help to solve this problem.it can save the classification model into a pickle file and persist the classification model for future use without having to retrain. Besides that, the pickle also can help to save the memory space.

Pickle is used for serializing and de-serializing Python object first before writing it to file. The serialization process refers to the process of converting an object in memory to a byte stream that can be stored on disk. And this character stream can be retrieved and de-serialized back to a Python object.

```

import pandas as pd
np.random.seed(500)
df = pd.read_csv('trainData.csv')
col = ['emotion', 'sentence']
df = df[col]
df = df[pd.notnull(df['sentence'])]
df['category_id'] = df['emotion'].factorize()[0]
category_id_df = df[['emotion', 'category_id']].drop_duplicates().sort_values('category_id')
category_to_id = dict(category_id_df.values)
id_to_category = dict(category_id_df[['category_id', 'emotion']].values)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df['sentence'], df['emotion'], test_size=0.1, random_state = 0)
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(sublinear_tf=True, min_df=5, max_df = 0.8, ngram_range=(1, 2))
X_features = tfidf.fit_transform(X_train).toarray()
print(X_features.shape)

(20198, 7565)

from sklearn.metrics import accuracy_score
from sklearn.svm import LinearSVC
svmclf = LinearSVC(C=0.25).fit(X_features, y_train)
y_pred = svmclf.predict(tfidf.transform(X_test))
print("Linear SVM Accuracy Score -> {:.2f}".format(accuracy_score(y_pred, y_test)*100))

from sklearn.naive_bayes import MultinomialNB
NBclf = MultinomialNB().fit(X_features, y_train)
y_pred1 = NBclf.predict(tfidf.transform(X_test))
print("Multinomial NB Accuracy Score -> {:.2f}".format(accuracy_score(y_pred1, y_test)*100))

from sklearn.ensemble import RandomForestClassifier
rfclf = RandomForestClassifier(n_estimators=100).fit(X_features, y_train)
y_pred2 = rfclf.predict(tfidf.transform(X_test))
print("rf Accuracy Score -> {:.2f}".format(accuracy_score(y_pred2, y_test)*100))

from sklearn.ensemble import BaggingClassifier
Brtclf = BaggingClassifier(RandomForestClassifier(random_state=1)).fit(X_features, y_train)
y_pred3 = Brtclf.predict(tfidf.transform(X_test))
print("ensemble Accuracy Score -> {:.2f}".format(accuracy_score(y_pred3, y_test)*100))

Linear SVM Accuracy Score -> 70.56
Multinomial NB Accuracy Score -> 67.66
rf Accuracy Score -> 71.76
ensemble Accuracy Score -> 71.54

```

Figure 5.1: Jupyter Notebook of sentiment analysis classification model

The Figure 5.1 is showing the Jupyter Notebook of sentiment analysis classification model. It is using clean train dataset and build four classification model to train dataset and provide the result.

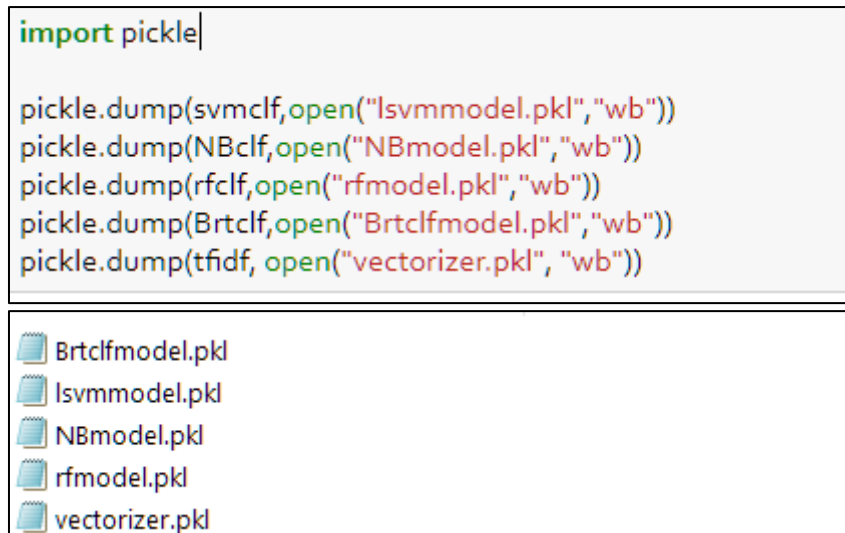


Figure 5.2; Build the pickle file.

In this project, it is needs five pickle file, which area the TfidfVectorizer, linear support vector, Naïve Bayes, random forest, and ensemble bagging with random forest. The TfidfVectorizer is use for vectorise the input data and other four classification is use for predict the input data that after process vectorise. After training the model, to import pickle to create the pickle as .pkl file for the later use (refers to the figure 5.2).

5.1.2 Backend

Flask



In order to develop a web-based sentiment analysis system, it is required to use the flask web framework. Flask is a popular python framework to help developers to create web applications. Flask is considered more pythonic than Django framework, this is because flask is more explicit. Flask will be used as a tool to build the backend application to perform the function.

As the beginner can refer this link: <https://realpython.com/tutorials/flask/>

In the python file need install some python packages that need to be used using pip.

1. Flask (python Web Framework)
2. Werkzeug (WSGI web application library)
3. Requests (handling requests)
4. natural language toolkit (text pre-processing)
5. Spacy (text pre-processing)
6. en_core_web_sm
7. emoji (text pre-processing)
8. unicodedata (text pre-processing)
9. pickle (load pkl file)

Newflask.py

```
import re
from flask import Flask, request, render_template
from werkzeug.utils import secure_filename
import os
import pandas as pd
import TextPreprocessing as tp
import classificationModel as clfmodel
from LVCdetection import detectLVC
from fixedExpressionDetection import detectFE
from idiomsDetection import detectidoms
from vpDetection import detectVp
import math
app = Flask(__name__)
```

```
@app.route("/")
def home():
    return render_template('main.html')
```

```
if __name__ == "__main__":
    app.run()
```

Figure 5.3: part of the newflask.py 1.

Implement

Newflask.py is contain the main code will be executed by the Spider to run the web application. It is include the file manage, text pre-processing, multiword expression detection, and classification model. It will initialized a new flask instance with the argument “_name_” to make flask can find the HTML temple folder in the same directory. Next, it is using “@app.route (“/”)” to specify the URL link that deliver to the main.html page, which is located the templates file.

```
@app.route('/result', methods = ['POST', 'GET'])
def result():
    if request.method == 'POST':
```

Figure 5.3: part of the newflask.py 2.

POST method

The “@app.route (“/result”, methods = [‘POST’, ‘GET’]))” is from the main.html page call the result.html page. It is access file upload from the users then it process the text pre-processing, multiword expression detection and classification model prediction.

The POST method is transport the form data to the server to process backend.

```

UPLOAD_FOLDER = 'uploaded_data'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

if request.method == 'POST':
    # take file from the user and save into uploaded file and output file
    f = request.files['file']
    f.save(os.path.join(app.config['UPLOAD_FOLDER'], secure_filename(f.filename)))
    fileName = re.sub('.txt', '', f.filename)
    tp.takefile('uploaded_data/' + f.filename, 'output_data/' + f.filename, 'output_data/mwe_' + fileName + '.txt' )

```

Figure 5.4: part of the newflask.py 3.

Save File

In the POST method, it is accept the form data from the users which is a text file. It using werkzeug to take the file save into uploaded folder and output folder in the same directory which can help to take the file to process the backend.

```

#take the file in the output file to process the mwe detection
mwe=""
if request.form.get('idm'):
    mwe = mwe+"Idioms"
    tp.process2('output_data/' + f.filename)
    detectidoms('output_data/' + f.filename, 'output_data/mwe_' + fileName + '.txt')

if request.form.get('fe'):
    mwe = mwe+ " Fixed Expression Construction "
    tp.process1('output_data/' + f.filename)
    detectFE('output_data/' + f.filename, 'output_data/mwe_' + fileName + '.txt')

if request.form.get('lvc'):
    mwe = mwe+" Light verb Construction"
    tp.process('output_data/' + f.filename)
    detectLVC('output_data/' + f.filename, 'output_data/mwe_' + fileName + '.txt')

if request.form.get('vp'):
    mwe = mwe+" Verb Particle "
    tp.process('output_data/' + f.filename)
    detectVp('output_data/' + f.filename, 'output_data/mwe_' + fileName + '.txt')
#if user no choose any mwe detection then it will process the normal text preprocessing
tp.process('output_data/' + f.filename)

```

Figure 5.5: part of the newflask.py 4.

Text pre-processing and Multiword Expression Detection

After save the file uploaded by users to the output folder, it take it to process text pre-processing and Multiword Expression detection choose by users.

```

# take the model choose to train data
modelchooseresult=""
modelchoose = request.form.get('modeltype')
if str(modelchoose) == 'lsvm':
    total,happiness,sadness, anger, fear,disgust , surprise = clfmodel.LSVMmodel_predict('uploaded_data/' + f.filename,'output_data/'+ f.filename, 'output_data/result_' + fileName + '.csv')
    modelchooseresult = 'Linear Support Vector Machine'
elif str(modelchoose) == 'nb':
    total,happiness,sadness, anger, fear,disgust , surprise = clfmodel.NBmodel_predict('uploaded_data/' + f.filename,'output_data/'+ f.filename, 'output_data/result_' + fileName + '.csv')
    modelchooseresult = 'Naive Bayes'
elif str(modelchoose) == 'rf':
    total,happiness,sadness, anger, fear,disgust , surprise = clfmodel.RFmodel_predict('uploaded_data/' + f.filename,'output_data/'+ f.filename, 'output_data/result_' + fileName + '.csv')
    modelchooseresult = 'Random Forest'
elif str(modelchoose) == 'ensemble':
    total,happiness,sadness, anger, fear,disgust , surprise = clfmodel.Brtmodel_predict('uploaded_data/' + f.filename,'output_data/'+ f.filename, 'output_data/result_' + fileName + '.csv')
    modelchooseresult = 'Ensemble (bagging) with Random Forest'

```

Figure 5.6: part of the newflask.py 5.

Model train

After text pre-processing and multiword expression detection, the data will be predict the emotion by using the selected classification model. The classification model after predict all data then count the total number of each 6 emotions.

```

#build the bar chart
labels = ['happiness', 'sadness', 'surprise', 'fear','anger','disgust']
values = [happiness,sadness , surprise, fear, anger, disgust ]
max1= max(values)
max1= roundup(max1)

```

Figure 5.7: part of the newflask.py 6.

Bar Chart

After classification model, it get total number of each 6 emotion. This number will store in the values then will bring the values the result.html page to process and generate the bar chart.

```

Result = pd.read_csv('output_data/result_'+fileName+'.csv')
return render_template("result.html",result = [Result.to_html(index=False)],resultfile = modelchooseresult,mweChoose=str(mwe),
    total1= total,happiness1 = happiness,sadness1 = sadness,anger1= anger, fear1= fear,disgust1 = disgust , surprise1 = surprise,
    title='Sentiment Analysis result', max=max1, labels=labels, values=values, steps=10 )

```

Figure 5.8: part of the newflask.py 7.

Result

After all backend process is done, then it bring all generate data and link to result.html with the generated data.

5.1.3 Text Pre-processing

TextPreprocessing.py

```
def clean_url(text):
    cleanr = re.compile('(https?://\w|\s)*(\www\.)?(\s)*((\w|\s)+\.)*([\\w|-]+\\.)*([\\w|-]+)(\\?)?(\\w|s)*=\\s*[\\w%&]*')
    cleantext = re.sub(cleanr, '', text)
    return cleantext

def clean_html(text):
    cleanr = re.compile('<.*?>|&([a-z0-9]+|#[0-9]{1,6}|#x[0-9a-f]{1,6});')
    cleantext = re.sub(cleanr, '', text)
    return cleantext

def clean_username(text):
    cleanr = re.compile('@[^\\s]+')
    cleantext = re.sub(cleanr, '', text)
    return cleantext

def strip_accents(text):
    return ''.join(c for c in unicodedata.normalize('NFD', text) if not unicodedata.combining(c))

def clean_punc(text):
    punctuation = "!@#$%^&*()_+=[{}]:\\|/<>?:.,;"
    for c in text:
        if c in punctuation:
            text = text.replace(c, ' ')
    return text

def replace_emoji(text):
    return emoji.demojize(text, delimiters=(" ", ""))

def processdata(text):
    text = clean_html(text)
    text = clean_url(text)
    text = clean_username(text)
    text = strip_accents(text)
    text = replace_emoji(text)
    text = clean_punc(text)
    # text = name_en_rec(text)
    text = tokenization(text)

def process(readFile,writeFile):
    conl=[]
    with open(readFile,encoding='utf-8') as txtfile:
        readtxt = csv.reader(txtfile, delimiter='\\n')
        for row in readtxt:
            row[0] = processdata(row[0])
            print(row[0])
            conl.append([row[0]])
    with open(writeFile, 'w', newline='', encoding='utf-8') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(conl)
```

Figure 5.9: part of TextPreprocessing.py

The figure 5.9 is showing the part of the TextPreprocessing.py. It is included the remove HTML tag, URL, @username, accents, emoji, punctuation, and using the tokenization to remove the stop words and convert text to lowercase and apply the lemmatization. After text pre-processing, the data will store back to same directory location

5.1.4 Multiword Expression implement

Light verb construction detection

LVCdetection.py

<pre>#import the spacy library import spacy import csv from spacy.matcher import Matcher #Load the dictionary model nlp = spacy.load('en_core_web_sm') matcher = Matcher(nlp.vocab) #rule based approach #declara the pattern pattern = [{ 'POS': 'VERB' }, { 'POS': 'DET', 'OP': '*' }, { 'POS': 'ADP', 'OP': '?' }, { 'POS': 'ADV', 'OP': '?' }, { 'POS': 'ADJ', 'OP': '*' }, { 'POS': 'NOUN', 'OP': '+' }] pattern1 = [{ 'POS': 'AUX' }, { 'POS': 'DET', 'OP': '*' }, { 'POS': 'ADP', 'OP': '?' }, { 'POS': 'ADV', 'OP': '?' }, { 'POS': 'ADJ', 'OP': '*' }, { 'POS': 'NOUN', 'OP': '+' }] # put the pattern into matcher matcher.add('light_verb', None, pattern) matcher.add('light_verb', None, pattern1)</pre>			
<pre>5 # tokenization each comment one by one 7 def data_processing(string): 8 # put the comment into spacy model 9 sentence = nlp(string) 10 #tokenization and lemmatization and lowercase the text 11 text = [token.lemma_.strip().lower() for token in sentence if token.lemma_ != '-PRON-' 12 and token.dep_ != "punct" and token.pos_ != "SYM" and token.is_digit==False and token.is_alpha == True] 13 #make it token become sentence 14 sentence_text = ' '.join(text) 15 return sentence_text</pre>			
<pre># Light verb detection def extract_full_name(string,filename): nlp_doc= nlp(string) #put the sentences matches = matcher(nlp_doc) for match_id, start, end in matches: verb = nlp_doc[start] span = nlp_doc[start:end] if verb.text == 'do' or verb.text == 'get' or verb.text == 'give' or verb.text == 'have' or verb.text == 'make' or verb.text == 'take': with open(filename) as csvfile: readCSV = csv.reader(csvfile, delimiter=',') for row in readCSV: if span.text == row[0]: string= string.replace(span.text,row[1]) print(string) csvfile.close return string</pre>			
<pre>def detectLVC(readFile,writeFile): result=[] with open(readFile,encoding='utf-8') as txtfile: readtxt = csv.reader(txtfile, delimiter='\n') for row in readtxt: data_lemma = data_processing(row[0]) # process the tokenization and Lemmatization data_convert = extract_full_name(data_lemma,'LVC.csv') # POS tagging by rule based approach technique result.append([data_convert]) #append the result writefile(result,writeFile)</pre>			
Original Sentence	Clean Sentence	Predicted Emotion	MWE
dad do take an hour to take a shower try have long curly thick ss hair be wash and shave w and he put on the thing	dad do take an hour to shower try have long curly thick ss hair be wash and shave w and put the thing	anger	LVC VP
He is angry and furious, takes a nap at the night club.l	be angry and furious sleep at the night club l	anger	LVC
i hope i can get bus on the other side	i hope i can get bus on the other side	sadness	
Bonnie had a walk	bonnie walk	surprise	LVC
Jasmine is sad with her life and wanting to take a break from the busy schedule.😞	jasmine be sad with life and want to break from the busy schedule smile face with heart eye	sadness	LVC

Figure 5.10:LVCdetection.py

The figure 5.10 is showing the light verb construction detection. This is located the output folder and take the sentences in the file to process the light verb construction detection. It is using Spacy library to load the en_core_web_sm model and created the matcher to use for rule based matching to detect the light verb construction pattern from the sentence. As the mention early, the common light verb construction is 'do', 'get', 'give', 'have', 'make', and 'take'. Therefore, the sentences is find the words match the pattern and it is one of the six common light verb construction then convert the light verb construction to the verb.

5.1.5 Classification Model

classificationModel.py

Linear Support vector Model

```
import pickle
import csv

svmlf = pickle.load(open("ModelPK1/lsvmmodel.pkl", 'rb'))
NBclf = pickle.load(open("ModelPK1/NBmodel.pkl", 'rb'))
rfclf = pickle.load(open("ModelPK1/rfmodel.pkl", 'rb'))
Brtclf = pickle.load(open("ModelPK1/Brtclfmodel.pkl", 'rb'))
tfidf = pickle.load(open("ModelPK1/vectorizer.pkl", 'rb'))

def countemotion(predict):
    total = 0
    happiness = 0
    sadness = 0
    anger = 0
    fear = 0
    disgust = 0
    surprise = 0
    for index in predict:
        if index == "happiness":
            happiness = happiness + 1
        elif index == "sadness":
            sadness = sadness + 1
        elif index == "anger":
            anger = anger + 1
        elif index == "fear":
            fear = fear + 1
        elif index == "disgust":
            disgust = disgust + 1
        elif index == "surprise":
            surprise = surprise + 1
    total = total + 1
    return total, happiness, sadness, anger, fear, disgust, surprise

def storeResult(org_sentences, clean_sentences, predict, mwe_sentences):
    result = []
    result.append(['Original Sentence', 'Clean Sentence', 'Predicted Emotion', 'MWE'])
    i = 0
    for index in predict:
        result.append([org_sentences[i], clean_sentences[i], predict[i], mwe_sentences[i]])
        i = i + 1
    return result

def LSVMmodel_predict(oriReadFile, readFile, writeFile, mwefile):
    org_source_document = open(oriReadFile, "r", encoding='utf-8').read()
    org_sentences = org_source_document.split("\n")

    clean_source_document = open(readFile, "r", encoding='utf-8').read()
    clean_sentences = clean_source_document.split("\n")

    mwe_document = open(mwefile, "r", encoding='utf-8').read()
    mwe_sentences = mwe_document.split("\n")

    predict = svmlf.predict(tfidf.transform(clean_sentences))

    result = storeResult(org_sentences, clean_sentences, predict, mwe_sentences)

    writefile(result, writeFile)

    return countemotion(predict)
```

Figure 5.11: par of classificationModel.py

As the mention above, the classification model is trained and save into as the pickle file. Therefore, in the backend process, the classification model is no necessary to train again. It is can be load the linear support vector model pickle model and the vectorizer into object. Then, using the classification model to predict the cleaned sentences to label the emotion. After prediction is done, then take original sentence, cleaned sentences and result to store into a csv file. The reason to take original sentence is use for show the different between the cleaned sentences. Besides that, after prediction, the result will process to count the total number of each 6 emotion and return back to newflask.py to use for build the bar chart.

5.1.6 Web Page

Main.html

```

<header>
<div class="headertop">
  <h1>Sentiment Analysis </h1>
</div>
</header>
<body>
<form action = "http://127.0.0.1:5000/result" method = "POST" enctype = "multipart/form-data">
<!-- To allow user choose a classification model -->
<div class="modelchoose">
  <label>Choose a Classification Model </label> <br/>
  <select id="modeltype" name="modeltype" required>
    <option value="lsvm">Linear Support Vector</option>
    <option value="nb">Naive Bayes</option>
    <option value="rf">Random Forest</option>
    <option value="ensemble">Ensemble (Bagging) with Random Forest</option>
  </select>
</div>
<!-- To allow user choose a Multiword Expression -->
<div class="mwechoose">
  <label>Choose the Multiword Expression Detection </label> <br/>
  <div>
    <input type="checkbox" id="idm" name="idm" value="idm"/> <label>Idioms</label><br/>
    <input type="checkbox" id="fe" name="fe" value="fe"/> <label>Fixed Expression</label><br/>
    <input type="checkbox" id="lvc" name="lvc" value="lvc"/> <label>Light Verb Construction</label><br/>
    <input type="checkbox" id="vp" name="vp" value="vp"/> <label>Verb Particle </label><br/>
  </div>
</div>

<div class="fileupload">
<!-- To allow user to upload the test file that consist the sentences -->
<label>Upload a File </label> <br/>
<input name="file" id="fname" type="file" accept=".txt" required/>
</div>

<div class="submitbtn">
<input type="submit" value="Run"/>
</div>
</form>
</body>

```

Figure 5.12: HTML code of the “main.html” page

The figure 5.12 is showing the HTML code of the main page of the sentiment analysis system. This page is create a form to record the users choose the classification model, Multiword Expression and the uploaded file. Then, it send to the “result.html” page with the record the data. Before link to the result page, the system will go to backend to generate the result.

result.html

```

<body>
<div class="back">
  <a href="http://localhost:5000/"> Back to Home </a>
</div>
<div class="toptitle">
  <h2>Result of {{resultfile}}</h2>
</div>
<div class="resultTable">
  <div class="filedetail">
    <div class="filedetail1">
      <div class="subdetail1">
        <label>Multword Expression detection</label><br><label style="font-weight:bold">{{mweChoose}}</label> <br><br>
        <label>Total Number of Data</label><br><label style="font-weight:bold">{{total1}}</label>
      </div>
      <div class="subdetail">
        <label>Number of Happiness</label><br><br><label style="font-weight:bold">{{happiness1}}</label> </div>
      <div class="subdetail">
        <label>Number of Sadness</label><br><br><label style="font-weight:bold">{{sadness1}}</label> </div>
      <div class="subdetail">
        <label>Number of Surprise</label><br><br><label style="font-weight:bold">{{surprise1}}</label> </div>
      <div class="subdetail">
        <label>Number of Fear</label><br><br><label style="font-weight:bold">{{fear1}}</label> </div>
      <div class="subdetail">
        <label>Number of Anger</label><br><br><label style="font-weight:bold">{{anger1}}</label> </div>
      <div class="subdetail">
        <label>Number of Disgust</label><br><br><label style="font-weight:bold">{{disgust1}}</label> </div>
    </div>
  </div>
</div>
<center>
  <h3>{{ title }}</h3>
</center>
<div style="border: 1px solid black; width: 550px;">
  <div style="border: 1px solid black; width: 550px height="300">
  </div>
</div>
<br><br>
<div class="filetable">
  {% for table in result %}
    {{ table|safe }}
  {% endfor %}
</div>
</div>
</body>

```

Sentiment Analysis

[Back to Home](#)

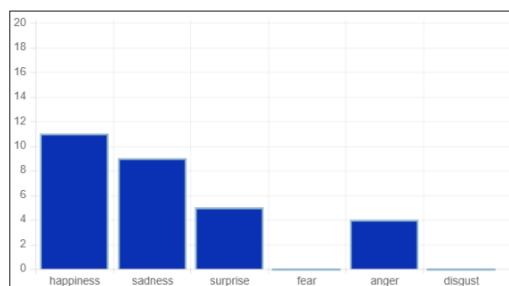
Result of Linear Support Vector Machine

Multword Expression detection
Idioms Fixed Expression Construction Light verb Construction Verb Particle

Total Number of Data
29

Number of Happiness	Number of Sadness	Number of Surprise	Number of Fear	Number of Anger	Number of Disgust
11	9	5	0	4	0

Sentiment Analysis result



Original Sentence	Clean Sentence	Predicted Emotion	MWE
i hope i can get bus on the other side	i hope i can get bus on the other side	sadness	
dad do take an hour to take a shower try have long curly thick ss hair be wash and shave w and he put on the thing	dad do take an hour to shower try have long curly thick ss hair be wash and shave w and put the thing	anger	LVC VP
The car that I bought recently was built by Koenigegg in Sweden and it can makes me feel cool and cheerful.	the car that i buy recently be build by koenigegg in sweden and can make feel cool and cheerful	sadness	

Figure 5.13: HTML code of the result page

The figure 5.13 is showing the HTML code of the result page of the sentiment analysis system. This page is created to display the result to users. After backend process, the data to bring to place of “{ {XXX} }” and display the information. The information is included the classification model, multiword expression detections choose, total number of each 6 emotions, bar chart and display the prediction result csv file in a table form.

5.1.7 Bar chart

```

<script src='https://cdnjs.cloudflare.com/ajax/libs/Chart.js/1.0.2/Chart.min.js'></script>
<div style="border: 1px solid black; width: 550px;">
  <canvas id="chart" width="550" height="300"></canvas>
</div>

<script>
  // bar chart data
  var barData = {
    labels : [
      {% for item in labels %}
        "{{ item }}",
      {% endfor %}
    ],
    datasets : [{
      fillColor: "rgba(10,49,179,1)",
      strokeColor: "rgba(151,187,205,1)",
      pointColor: "rgba(151,187,205,1)",
      data : [
        {% for item in values %}
          "{{ item }}",
        {% endfor %}
      ]
    }
  ]

  // get bar chart canvas
  var mychart = document.getElementById("chart").getContext("2d");
  steps = {{steps}}
  max = {{max}}
  // draw bar chart
  new Chart(mychart).Bar(barData, {
    scaleOverride: true,
    scaleSteps: steps,
    scaleStepWidth: Math.ceil(max / steps),
    scaleStartValue: 0,
    scaleShowVerticalLines: true,
    scaleShowGridLines : true,
    barShowStroke : true,
    scaleShowLabels: true
  });
</script>

```

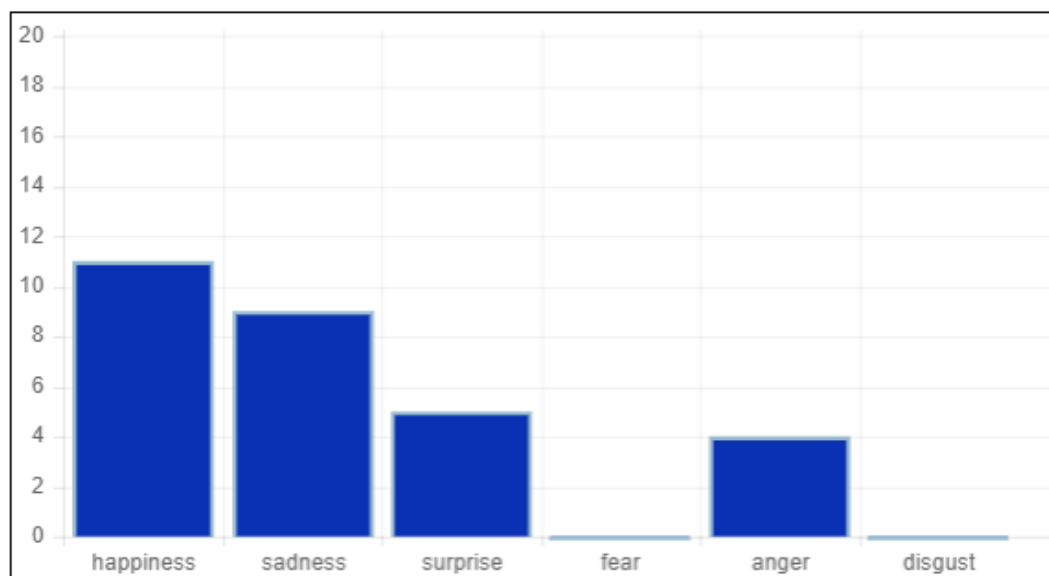


Figure 5.13: HTML code of the result page

In order to build a bar chart, it is need to call the `char.js` file in the script of `result.html` file. The `chart.js` is a popular and powerful data visualization library to build the chart, such as bar chart, line chart, pie chart and other. The `chart.js` library add its flexibility for customize the plots by feeding the certain parameters, which can be build a nice chart. To created a canvas nodes to render the chart and specify the width and height of the chart. Then, create a java script to customize the colour of bar chart and get values and item from the backend to make a bar chart.

5.1.8 Upload to CentOS server

After program is done, the project team members is uploaded the program to CentOS server by using the winscp. CentOS is the Linux server and it allow the program process faster. First, the team members is using the FortiClient VPN and configure the pre-shared key and connect VPN by using the `ddamin` account. Then, using the remote desktop connection to connect 10.123.51.22 ip in order to connect the CentOS server. Next, team member was installed anaconda 3 and using this to open this program and run.

5.2 Testing

5.2.1 Unit Testing

The unit testing will be defined as the testing method where individual component of the system were tested. The unit testing can used to find error in the early stage of development of the system. The purpose of unit testing will be validated each unit of the system perform as the function and ensure that no error was found within the individual component before it integrates with other components.

5.2.2 Integration Testing

The integration testing will be test the combined individual components as a group after unit testing. The main goal of the conducting this testing is to evaluate the compliance of the system with specified functional requirements. The purpose of integrate testing is to expose faults in the interaction between integrated units.

5.2.3 System testing

The system testing is the software testing process to testing the completed and integrated system. The purpose of the system testing is to evaluate the system compliance against the requirement specification.

5.2.4 Test Case

Test Case ID: T_01	Test Case Name: testing the classification model
System: Sentiment Analysis System	Subsystem: classification model
Designed By: Liew Er Wei	Design Date: 29-10-2020
Executed By: Liew Er Wei	Execution Date:
Short description: to testing the classification can work well.	

Pre-conditions: link the main page

Step	Action	Test Data	Expected Result	Pass / Fail	Remark
1	Select linear support vector model	testing.txt	Get prediction result	Pass	
2	Select Naïve Bayes model	testing.txt	Get prediction result	Pass	
3	Select random forest model	testing.txt	Get prediction result	Pass	
4	Select ensemble bagging with random forest model	testing.txt	Get prediction result	Pass	

Post-conditions: Users get the prediction result

Test Case ID: T_02	Test Case Name: light verb construction detection
System: Sentiment Analysis System	Subsystem: light verb construction detection
Designed By: Liew Er Wei	Design Date: 29-10-2020
Executed By: Liew Er Wei	Execution Date:
Short description: to testing the light verb construction can work well	

Pre-conditions: link to main page

Step	Action	Test Data	Expected Result	Pass / Fail	Remark
1	Tick the light verb construction detection	Testing_LVC.txt	The light verb construction is detected and convert it to verb	Pass	
2	Tick the light verb construction detection and other type of Multiword Expression detection	Testing_LVC.txt	The light verb construction is detected and convert it to verb	Pass	

Post-conditions: The cleaned sentences is show the light verb construction is convert to verb and the sentences to become shorter.

5.3 Testing

Testing_LVC.txt (Light verb construction detection)

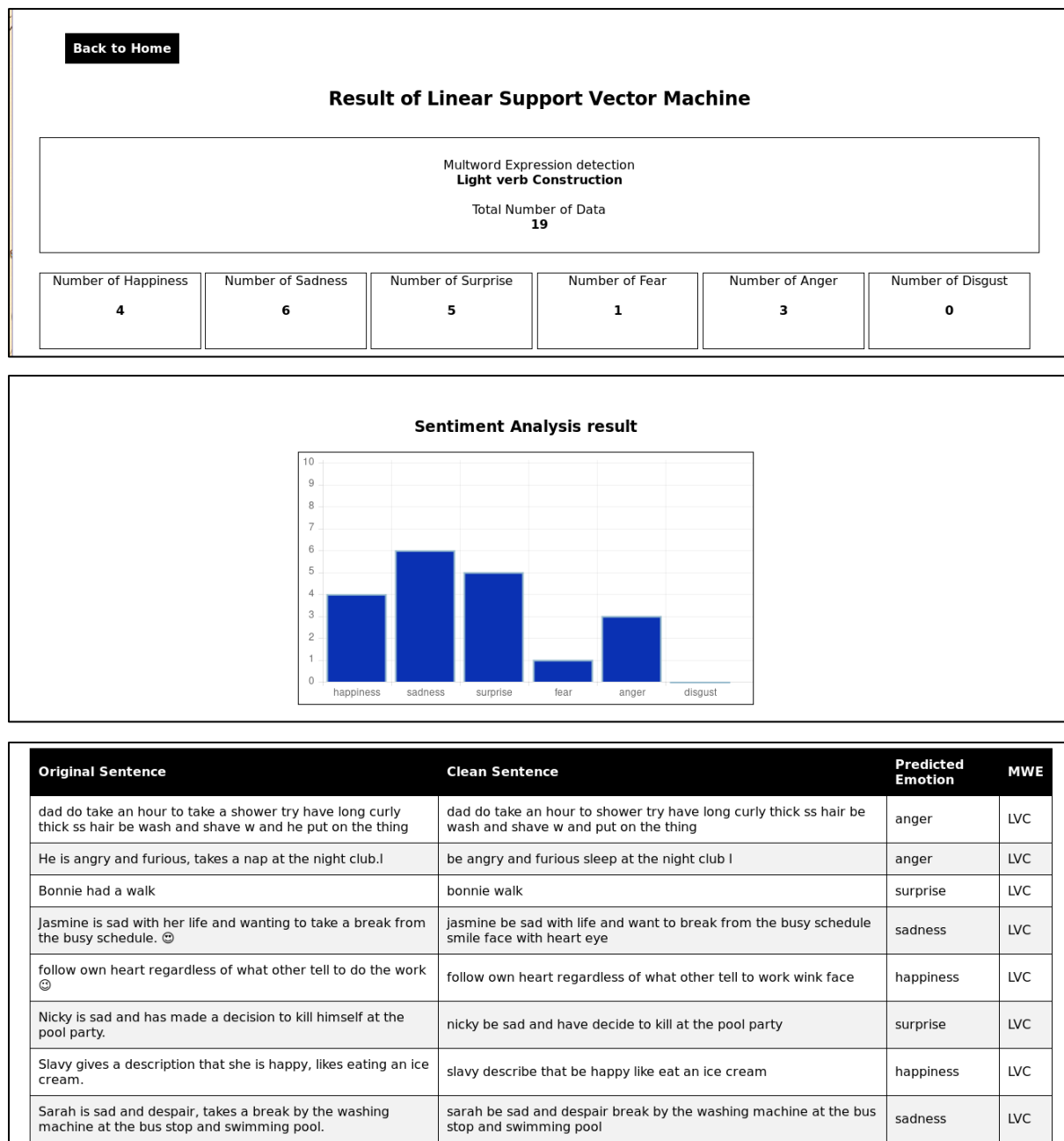
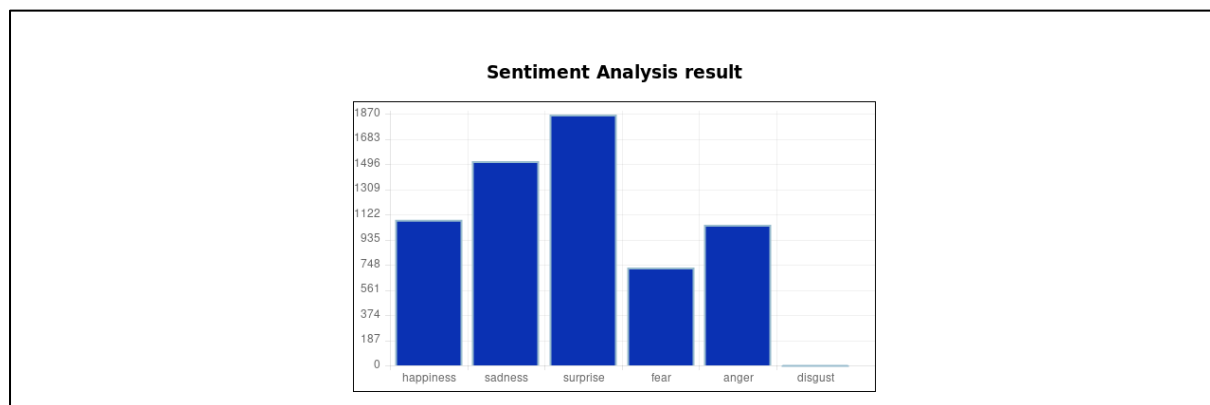


Figure 5.14: result of the testing_LVC.txt

The figure 5.14 is showing the result of texting_LVC.txt produce by the sentiment analysis system. Based on the table in figure 5.14, found that the MWE column show the sentences was consist of light verb construction and the system have detected it and convert it to verb.

Covid19.txt (all multiword expression detection)

Result of Linear Support Vector Machine					
Multword Expression detection Idioms Fixed Expression Construction Light verb Construction Verb Particle Total Number of Data 6238					
Number of Happiness	Number of Sadness	Number of Surprise	Number of Fear	Number of Anger	Number of Disgust
1082	1518	1863	726	1045	4



Original Sentence	Clean Sentence	Predicted Emotion	MWE
text	text	happiness	
@molo_i_poka I'd rather drink alcohol as opposed to listening to all these pastor claiming they can cure Covid-19.Â?Â?Â?Â?Â?Â? I am sure we can all pray in the comfort of our homes.	molo_i_poka i rather drink alcohol as oppose to listen to all these pastor claim can cure covid a a a a a i be sure can all pray in the comfort of home	happiness	
"@MbuyiseniNdlozi There's an alcohol in every sanitizers while covid 19 hits the internal surfaces of human organs.	mbuyisenindlozi there be an alcohol in every sanitizer while covid hit the internal surface of human organ let see if consume the real sanitizer the same as apply every time on hand a daily basis i think drink alcohol be the cure	surprise	VP
Let us see if we consume the real sanitizers the same as we apply it every time on our hands on a daily basis.	let be responsible when alcohol be available drinking alcohol be not covid cure but can lead infection if not responsible levellockdown	surprise	VP
I think drinking alcohol is the cure."	excuse do alcohol cure covid shld b focus on vaccine developmental stag as anticipate a spike in infection but some celebrate can consume alcohol but remain prone to the virus give more reason to hate alcohol	anger	

Figure 5.15: result of the covid19.txt

The figure 5.14 is showing the result of covid19.txt produce by the sentiment analysis system. This dataset is consist 6238 sentences and it is related opinion with the Covid 19. Based on the sentiment analysis result, it show that more opinion is consist surprise and sadness emotion in the sentences. It may be the people was surprise that the Covid 19 is series cases and people was feel sad due the victim of Covid 19. Besides that, this result also consist the four type of the multiword expression detection.

5.4 Chapter Summary and Evaluation

In the development, the system is allow the machine learning to predict the sentiment in each sentences. Since the result in chapter 3 is prove that the Multiword Expression can improve the performance of machine learning approach. Therefore, the sentiment analysis system is combine with the Multiword Expression in order to provide more accurate result to the user. In the testing, all of testing results of the system is achieve the expected results. Therefore, the system is meet the requirement and it is perform correctly.

Chapter 6

Multiword Expression Toolkit

And

Ngram Statistic Package

6. MWEToolKit and

6.1 Multiword Expression Toolkit (MWEToolKit)



Figure 6.1: logo of mwetoolkit

The Multiword Expression Toolkit (MWEToolKit) is a tool that aids in the automatic identification of the multiword unit from the corpus. The MWEToolkit is developed and maintained by Silivio Ricardo Cordeiro and Carlos Ramisch. The MWEToolKit can provide a target list of the MWE candidates. These lists are extracted and filtered based on the number of user-defined criteria and a set of standard statistical association measures. In order to generate corpus counts, the MWEToolKit provides both corpus indexing tools and tools integrated with Web search engines. For evaluation, it provides validation and annotation tools. Besides that, The MWEToolKit also allows integrate with the machine learning approach for the creation and application of the supervised MWE extraction model to used to predict the text documentation. The main goal of mwetoolkit is to provide targeted candidate Multiword expressions (MWE) and Multiword tokens (MWT) lists for lexicographers and temperature recorders to speed up the creation of general and professional dictionaries. Based on the research, the MWEToolKit was used by researchers to tested and evaluated in the context of MWE extraction in the biomedical domain. As the result, the MWEToolKit performs better than other approaches, especially concerning recalls.

6.1.1 Feature of Multiword Expression Toolkit (MWEToolKit)

1. MWEToolKit efficiently perform multilevel regular expression search in the large corpus.
2. MWEToolKit can efficiently perform n-gram and word counting in large corpora.
3. MWEToolKit can associate measures, contrastive measures, variations, other.
4. MWEToolKit is tool for formatting, preprocessing, combining text statement information.
5. MWEToolKit can mark instance comments through lexicon and pattern projection.

6.1.2 Installation

The installation of the MWEToolKit will be different based on different operating system (OS). There are three method for three operating system, which are the Linux, Mac OS and Windows. After install the MWEToolKit need download the GIT through Cygwin. GIT is a version control system, then it can help the users to install or upgrade the MWEToolKit to latest version at any time.

6.1.3 Sequence diagram

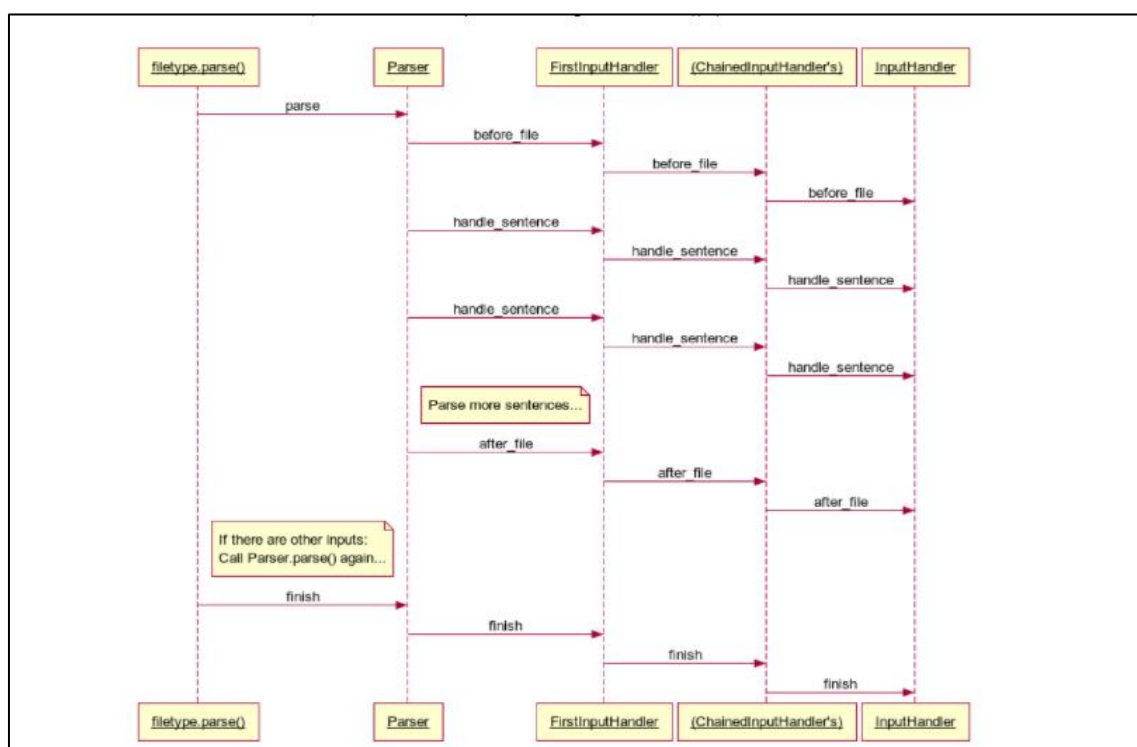


Figure 6.2: sequence diagram of the MWEToolKit.

6.1.4 System Architecture

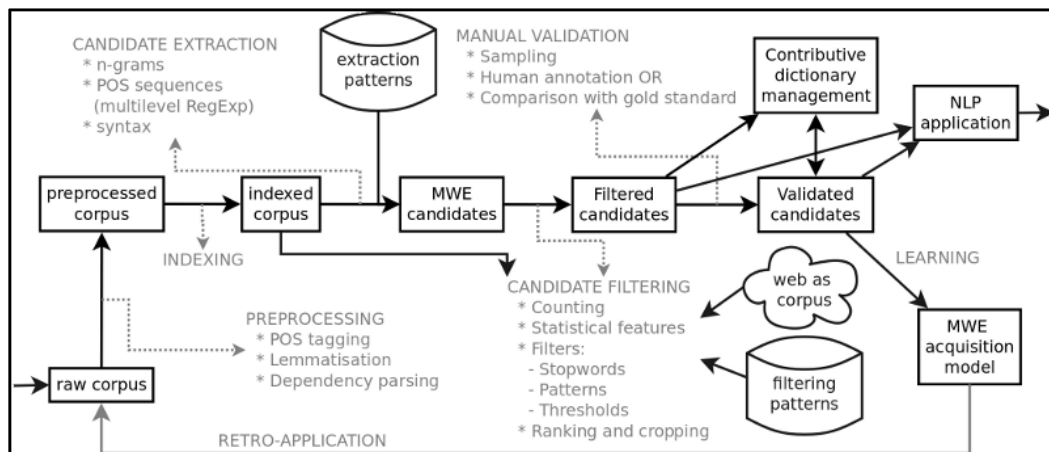


Figure 6.3: Architecture scheme of the MWEToolKit.

The figure 6.3 is showing the architecture scheme of the MWEToolKit. The MWEToolKit is designed as a set of Python scripts to process intermediate representation of the corpus, MWE pattern list, MWE candidate list and the reference dictionary. From the original corpus to the filtered MWE candidate list, each script performs a specific task in the MWE extraction process, and if a reference gold standard is given, it includes their automatic evaluation.

6.1.5 Component of MWEToolKit

1. `annotate_crf.py` - CRF model learned with classification model
2. `annotate_mwe.py` - Take a corpus and MWE lexicon and output
3. `avg_precision.py` - Take a MWE lexicon and provide a average precision summary
4. `candidates.py` - Apply a pattern to corpus and provide the MWE lexicon candidates
5. `change_pos.py` - Convert the corpus POS tagging and provide new corpus
6. `combine_freqs.py` - combine the frequency of multiple sources in the MWE lexicon
7. `content_words.py` - remove all words that are not noun, verb, adjective from the corpus
8. `counter.py` - calculate the frequency of the each word in a MWE lexicon
9. `cycle.py` - cycle through all input entries
10. `eval_manual.py` - command-line interface to manually evaluate the MWE lexicon
11. `feat_contrast.py` - read a MWE lexicon annotated with frequency from multiple sources.
12. `Filter.py` - filter entries in an input file
13. `Fix_feature.py` - correct inexistent feature values

6.2 Ngram Statistic Packages

The Ngram Statistic Package (NSP) can also known as Text::NSP. It is created by Ted Pedersen, Satanjeev Banerjee, Amruta Purandare, Saiyam Kohli. It is a free available open source software that to identifies the ngrams, collocation and word association in the text sentences. It is implemented in the Perl and it is helpful of the regular expressions to provide more flexible tokenization. Besides that, it is allow for the identification of non-adjacent ngrams and it have included a wide range of the measure of association that can be used to identify the collocation.

Identification of the multiword expressions (MWEs) is a challenge in natural language processing (NLP). Therefore, the availability of flexible and the easy to use toolkit is important to help in identification of the MWE. One of the toolkit is Text::NSP, it is a packages and included program for counting ngrams, measure the association between the words that make up ngrams and measure the correlation between the rankings of the ngrams. Ngram is an ordered sequence of token from the text sentences.

6.2.1 Feature of Ngram Statistic Packages

1. NSP can easy to perform the Tokenization of text file.
2. NSP can be assembles sequences of token into Ngrams
3. NSP can help counting Ngram frequencies which included counting Bigrams, counting Ngram, Ngram filters.
4. NSP can measures of the association for Ngrams
5. NSP can help to comparing ranked list o Ngrams

6.2.2 Installation

Before install can through this link to download the packages:
<https://cpan.metacpan.org/authors/id/T/TP/TPEDERSE/Text-NSP-1.31.tar.gz>. In order to install the text::NSP, it is require installed on the system with Perl. Next, the system need to have superuser root access. Then, follow instruction below.

```
perl Makefile.PL
make
make test
su          # or sudo, as the case may be
make install

cd Testing # optional, but recommended
csh ./ALL-TESTS.sh
cd ..

make clean
```

Figure 6.4: instruction for installation of text::NSP

6.2.3 Class Diagram

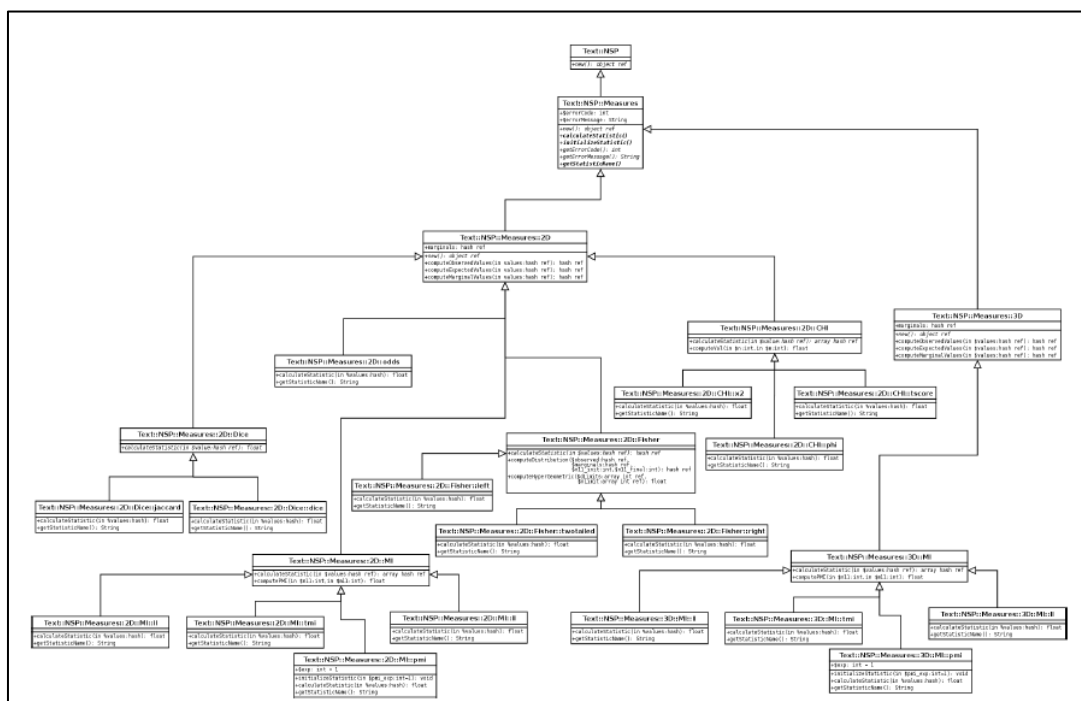


Figure 6.4: class diagram of text::NSP

6.2.4 Component

Found that in text::NSP there are many component. However, based on the research, there a few component helpful in identification of Multiword Expressions.

1. **count.pl**

The count.pl program is take any number of plain text file or sentence of and counts the total number of ngrams. It is provide the tokenization function to define what a token may be using the regular expression. Although it usually takes more time, this can be scaled to a larger amount of text. Before running count.pl, the combig.pl program have to adjust the count from count.pl to reflect the unordered count. Then, the find-compounds.pl program will be specify a file of known multiword expressions, and then recognize all these occurrences in the corpus.

2. **statistic.pl**

The text::NSP is a wide range of association metrics that can be used to identify interesting ngrams, especially bigrams and trigrams. These measures are divided into families with common characteristics. This allows an object-oriented implementation, thereby facilitating the inheritance of common functions between these measures

3. **rank.pl**

The rank.pl program is takes the output of the statistic.pl program of two different metrics as input, and calculates the Spearman's Rank Correlation Coefficient between them. This can help to identify when two measures are very similar or different.

4. **kocos.pl**

The kocos.pl program can construct a word network by finding all n-order co-occurrences of a given text or regular expressions. This can be viewed recursively to some extent, where the third-order co-occurrence of a given target word is all the tokens that appear with the second-order co-occurrence.

6.3 Chapter Summary and Evaluation

In this chapter, it is describe two external open source software, which the multiword expressions toolkit (MWEToolKit) and Ngram Statistic Package (NSP). These software can be to help in identification of Multiword Expressions. Besides that, in this chapter have described a few component of those software. It found that, there are many way used to identify the multiword expression. However, make the comparison, found the multiword expression toolkit (MWEToolKit) is more efficient to identification of Multiword Expressions, this is because that, the NSP is require to perform complexity way to set up the Ngrams to help to match the Multiword Expression with the correct POS tagging to token the Ngrams.

Chapter 7

Discussions and Conclusion

7. Discussions and Conclusion

In this chapter, it will discussed the project's summary, achievement and contribution made by the system, limitation and future improvement, issue and solution.

7.1 Summary

The emotion will be affect our quality of life and the way we interact with other. They determine our thinking, the action we take, our subjective understanding of the world, and our behavioral responses. Therefore, the sentiment analysis system is helpful to help the people to analysis and predict their emotion. However the existence of the multiword Expression will become a key problem face by the sentiment analysis system. This is because it may consist more complicated and it contain different semantics to express certain emotions. I this process, it may affect the result of the sentiment analysis. Besides that, based on the research, there are many supervised machine learning algorithm suitable use for sentiment analysis. Therefore, the project team member was compare the three supervised machine learning algorithm to choose the best model to use in the sentiment analysis system. In addition, each project team member have different task of the multiword expression detection. The multiword expression detection is used for improve the performance of the sentiment analysis. Based on the research, found that the light verb construction is very flexible, this is because it is very complicated due to the widespread use of adjectives and adverb. This is problem face in the light verb construction detection. In order to solve this problem, it is using the rule based matching to match the light verb construction pattern. The rule based matching is helpful in the light verb construction to make the system more success to detect the light verb construction in the sentences and replace to verb. This can make sentences shorter, then the sentiment analysis system can easier to target the opinion words to predict more correct emotion.

7.2 Achievements

In this project is successfully fulfilled the primary objective of this project as develop a web based sentiment analysis system by detecting the Multiword Expression. Besides that, this project also successful prove that the Multiword Expression detection can be improve the performance of the sentiment analysis. The sentiment analysis allow using the Multiword Expression detection to detect the four type of the Multiword Expression in the sentences, which are includes the light verb construction, verb particle construction, fixed expression and the idioms construction.

7.3 Contributions

Most companies are always overwhelmed by the need for Sentiment Analysis systems. This is because the system can help them analyze and detect the sentiment in the comment text sentences obtained from the social media platform. By using this system, the company can understand the customer's satisfaction with its products or services. Examples include anger, disgust, fear, happiness, sadness and surprise. With this information, the company can make necessary changes or improvements to its products or services. As a result, companies can increase their customers' satisfaction and acceptance to improve business performance. In addition, the system can detect and manage multiword expressions in the comment. For example, light verb construction. The light verb construction extraction makes the sentences shorter in order to allow the sentiment analysis system easier to target the opinion words to decide the emotion. Therefore, the detection of Multiword Expression can help the system improve the accuracy of sentiment analysis.

7.4 Limitations and Future Improvements

Limitations

The limitation of this project is limited type of Multiword Expression. In this project, it is only consist four type of the Multiword Expression. Second, the project may is use to perform English sentences. However, there are many different language or consist more than one language sentences found in Malaysia.

Further Improvement

The future improvement of the project can add more type of Multiword Expressions. There are found that there are type of Multiword Expressions, therefore, it can add more Multiword Expression to make the Multiword Expression detection more complete. The second future improvement is can add other type of sentiment analysis, such as the speech recording, face detection and other. This can allow to perform various type of sentiment analysis. Third, the sentences in Malaysia may be the cross-lingual sentences, which mean that the sentences consist more than one language. Therefore, the sentiment analysis can using the machine translation to make the sentiment analysis more successful.

7.5 Issues and Solutions

In this project development, there were several issues encountered by the team member. Firstly, the first issues is lack of the knowledge and experience with sentiment analysis and multiword Expression. This is because team member was lack of understanding about the sentiment analysis and the multiword Expression. Therefore, a lot online research had been done in order to figure out the most suitable approach to perform the sentiment analysis and the multiword Expression, such as the how the sentiment analysis perform, type of the multiword Expression, various text preprocessing and others.

Secondly, the lack of the data set of the Multiword Expression was one of the issues faced during the project development. In order to detect the Multiword Expression, it is necessary the Multiword Expression dataset. However, the internet is lack of multiword Expression dataset. Therefore, the team member was collect the dataset from the online website and manually add record into the dataset.

Thirdly, the team member was hard to communicate to other due to the MCO period. Due to the Coronavirus (Covid 19), the team member cannot gather to discuss about the project and to interact with other. Before the MCO period, the team member can be easy gather in a room to discuss the project and share opinion to other. Therefore, the team member was use the google meet to perform online communication in order successful complete the project

8. References

Shah, H, 2018. Twitter Sentiment Analysis. *International Journal of Advanced Research in Computer Science and Software Engineering*, 7(12), p.15.

Bălan, O., Moise, G., Petrescu, L., Moldoveanu, A., Leordeanu, M. and Moldoveanu, F, 2019. Emotion Classification Based on Biophysical Signals and Machine Learning Techniques. *Symmetry*, 12(1), p.21.

Alnawas, A. and Arıcı, N, 2018. The Corpus Based Approach to Sentiment Analysis in Modern Standard Arabic and Arabic Dialects: A Literature Review. *Journal of Polytechnic*.

Tripathy, A. and Rath, S.K, 2017. Classification of Sentiment of Reviews using Supervised Machine Learning Techniques. *International Journal of Rough Sets and Data Analysis*, 4(1), pp.56–74.

Choudhari, P. & S, V.D. 2017, "Sentiment Analysis and Machine Learning Based Sentiment Classification: A Review", *International Journal of Advanced Research in Computer Science*, vol. 8, no. 3.

Ghiassi, M. and Lee, S, 2018. A domain transferable lexicon set for Twitter sentiment analysis using a supervised machine learning approach. *Expert Systems with Applications*, 106, pp.197–216.

Chang, G. and Huo, H, 2018. A METHOD OF FINE-GRAINED SHORT TEXT SENTIMENT ANALYSIS BASED ON MACHINE LEARNING. *Neural Network World*, 28(4), pp.325–344.

S., R., V., P., S., S. and Nagpal, D, 2018. Twitter Data Sentiment Analysis and Visualization. *International Journal of Computer Applications*, 180(20), pp.14–16.

Arulmurugan, R., Sabarmathi, K.R. and Anandakumar, H, 2017. Classification of sentence level sentiment analysis using cloud machine learning techniques. *Cluster Computing*.

Prabhat, A. and Khullar, V, 2017. Sentiment classification on big data using Naïve bayes and logistic regression. [online] IEEE Xplore. Available at: <https://ieeexplore.ieee.org/document/8117734> [Accessed 28 Jun. 2020]

Ramadhan, W.P., Astri Novianty, S.T.M.T. and Casi Setianingsih, S.T.M.T, 2017. Sentiment analysis using multinomial logistic regression. [online] IEEE Xplore. Available at: <https://ieeexplore.ieee.org/document/8226700> [Accessed 28 Jun. 2020].

Shubham, D., Mithil, P., Shobharani, M. and Sumathy, S, 2017. Aspect level sentiment analysis using machine learning. IOP Conference Series: Materials Science and Engineering, 263, p.042009.

Lee, C. and Chau, D, 2018. Language as pride, love, and hate: Archiving emotions through multilingual Instagram hashtags. *Discourse, Context & Media*, 22, pp.21–29.

Jain, S. and Asawa, K, 2015. EMIA: Emotion Model for Intelligent Agent. *Journal of Intelligent Systems*, 24(4).

Bălan, O., Moise, G., Petrescu, L., Moldoveanu, A., Leordeanu, M. and Moldoveanu, F, 2019. Emotion Classification Based on Biophysical Signals and Machine Learning Techniques. *Symmetry*, 12(1), p.21.

Fotopoulou, Aggeliki & Giouli, Voula, 2018. MWEs and the Emotion Lexicon: Typological and cross-lingual considerations. In *Multiword expressions: Insights from a multi-lingual perspective*. Berlin: Language Science Press, pp. 63–91.

Constant, M., Eryiğit, G., Monti, J., van der Plas, L., Ramisch, C., Rosner, M. and Todirascu, A, 2017. Multiword Expression Processing: A Survey. *Computational Linguistics*, 43(4), pp.837–892.

Ronan, P. and Schneider, G, 2015. Determining light verb constructions in contemporary British and Irish English. *International Journal of Corpus Linguistics*, 20(3), pp.326–354.

Vaidya, A., Agarwal, S. and Palmer, M, 2016. Linguistic features for Hindi light verb construction identification. [online] pp.1320–1329. Available at: <https://www.aclweb.org/anthology/C16-1125.pdf> [Accessed 20 Jun. 2020].

Wirth, R. and Hipp, J, n.d. CRISP-DM: Towards a Standard Process Model for Data Mining. [online] Available at: <http://www.cs.unibo.it/~danilo.montesi/CBD/Beatriz/10.1.1.198.5133.pdf>.

Kolchyna, O., Souza, Tharsis T. P, Treleaven, P. and Aste, T, 2015. Twitter Sentiment Analysis: Lexicon Method, Machine Learning Method and Their Combination. [online] arXiv.org. Available at: <https://arxiv.org/abs/1507.00955>.

B.Chrysal, J. and Joseph, S, 2015. Multi-Label Classification of Product Reviews Using Structured SVM. International Journal of Artificial Intelligence & Applications, [online] 6(3), pp.61–68. Available at: <http://aircconline.com/ijaia/V6N3/6315ijaia06.pdf> [Accessed 5 Jul. 2019].

Salehi, B., Cook, P. and Baldwin, T, 2015. A Word Embedding Approach to Predicting the Compositionality of Multiword Expressions. [online] Association for Computational Linguistics, pp.977–983. Available at: <https://www.aclweb.org/anthology/N15-1099.pdf> [Accessed 6 Jul. 2020].

Gharbieh, W., Bhavsar, V. and Cook, P, 2017. Deep Learning Models For Multiword Expression Identification. [online] Association for Computational Linguistics, pp.54–64. Available at: <https://www.aclweb.org/anthology/S17-1006.pdf> [Accessed 6 Jul. 2020].

Farahmand, M., & Nivre, J, 2015. Modeling the Statistical Idiosyncrasy of Multiword Expressions (pp. 34–38). Retrieved from Association for Computational Linguistics website: <https://www.aclweb.org/anthology/W15-0905.pdf>

Cambria, E., Poria, S., Gelbukh, A. and Thelwall, M, 2017. Sentiment Analysis Is a Big Suitcase. IEEE Intelligent Systems, 32(6), pp.74–80.

Rodrigues, I, 2020. CRISP-DM methodology leader in data mining and big data. [online]. Available at: <https://towardsdatascience.com/crisp-dm-methodology-leader-in-data-mining-and-big-data-467efd3d3781> [Accessed 2 Jul. 2020].

Kumar, S., Behera, P. and Jha, G.N, 2017. A classification-based approach to the identification of Multiword Expressions (MWEs) in Magahi Applying SVM. *Procedia Computer Science*, 112, pp.594–603.

Wehrli, E, 2017. Measuring the impact of collocational knowledge on sentence parsing. *Kāñina*, 40(4), p.49.

Nenonen, Marja & Mulli, Juha & Nikolaev, Alexandre & Penttilä, Esa, 2017. How light can a light verb be? Predication patterns in V + NP constructions. [online] Available at: https://www.researchgate.net/publication/317433559_How_light_can_a_light_verb_be_Predication_patterns_in_V_NP_constructions [Accessed 3 Jul. 2020].

Bak, J. n.d.. The Light Verb Construction in Korean. [online] Available at: https://tspace.library.utoronto.ca/bitstream/1807/31684/1/Bak_Jaehee_201111_PhD_thesis.pdf [Accessed 12 Jul. 2020].

Chen, W.-T., Bonial, C. and Palmer, M, 2015. English Light Verb Construction Identification Using Lexical Knowledge. [online] www.aaai.org. Available at: <https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/viewPaper/9925> [Accessed 20 Jul. 2020].

Ronan, P. and Schneider, G, 2015. Determining light verb constructions in contemporary British and Irish English. *International Journal of Corpus Linguistics*, 20(3), pp.326–354.

9. Appendices

Appendix 1: User Guide

System Document:

Operating system: Microsoft Window.

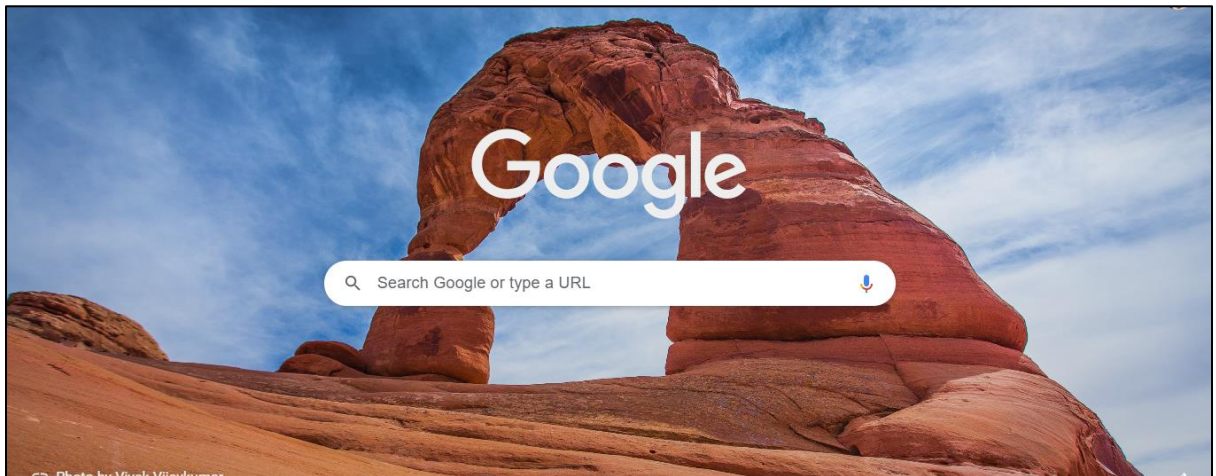
Software: Browser.

Hardware: Computer.

Framework: Flask web framework.

Operation Document

1. Open Brower



2. Link to the Main page

A screenshot of a web application titled "Sentiment Analysis". The interface has a black header with the title in white. Below the header, there is a section titled "Choose a Classification Model" with a dropdown menu currently showing "Linear Support Vector". Underneath, there is a section titled "Choose the Multiword Expression Detection" with four unchecked checkboxes: "Idioms", "Fixed Expression", "Light Verb Construction", and "Verb Particle". Further down, there is a section titled "Upload a File" with a "Choose File" button and the text "No file chosen". At the bottom, there is a black button labeled "Run".

3. Choose the classification model and multiword expression detection

4. Click upload file and choose a file to upload to web page

Sentiment Analysis

Choose a Classification Model

Linear Support Vector

Choose the Multiword Expression Detection

- ☒ Idioms
- ☒ Fixed Expression
- ☒ Light Verb Construction
- ☒ Verb Particle

Upload a File

Choose File testing.txt

Run

5. Click the run to perform the sentiment analysis and receive the result.

Sentiment Analysis

[Back to Home](#)

Result of Linear Support Vector Machine

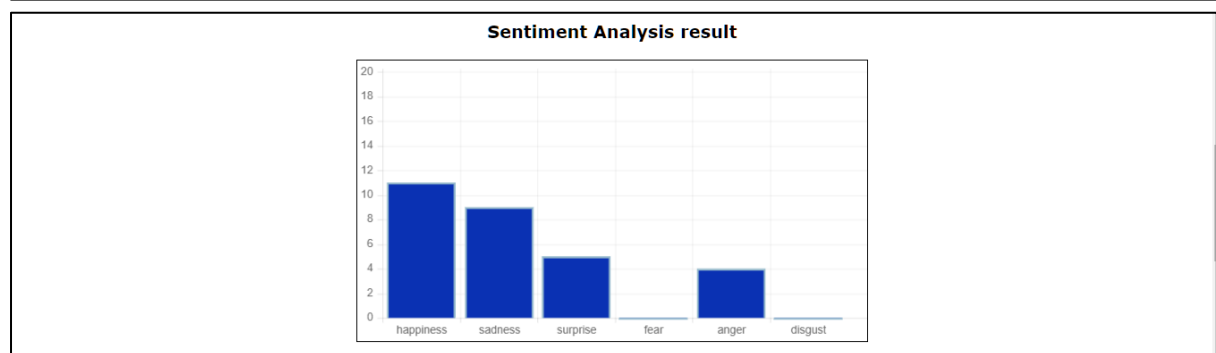
Multword Expression detection

Idioms Fixed Expression Construction Light verb Construction Verb Particle

Total Number of Data

29

Number of Happiness	Number of Sadness	Number of Surprise	Number of Fear	Number of Anger	Number of Disgust
11	9	5	0	4	0



Original Sentence	Clean Sentence	Predicted Emotion	MWE
i hope i can get bus on the other side	i hope i can get bus on the other side	sadness	
dad do take an hour to take a shower try have long curly thick ss hair be wash and shave w and he put on the thing	dad do take an hour to shower try have long curly thick ss hair be wash and shave w and put the thing	anger	LVC VP
The car that I bought recently was built by Koenigegg in Sweden and it can makes me feel cool and cheerful.	the car that i buy recently be build by koenigegg in sweden and can make feel cool and cheerful	sadness	
follow own heart regardless of what other tell to do the work😄	follow own heart regardless of what other tell to do the	happiness	

Appendix 2: Developer Guide

Database Required

1. Train Sentiment data
2. Light Verb Construction Data
3. Fixed Expression Data
4. Verb Particle Data
5. Idioms Data

File Required

1. Newflask.py
2. TextPreprocessing.py
3. idiomsDetection.py
4. LVCdetection.py
5. fixedExpressionDetection.py
6. vpDetection.py
7. vectorizer.pkl
8. rfmodel.pkls
9. NBmodel.pkl
10. lsvmmodel.pkl
11. Brtclmodel.pkl
12. Main.html
13. Result.html

The library required

1. numpy
2. pandas
3. spacy
4. WordNet
5. Flask
6. En_core_web_sm
7. Nltk
8. Werkzeug
9. Requests

This page is intentionally left blank to indicate the back cover. Ensure that the back cover is black in color.