



# 臺北城市科技大學

資訊管理系電子商務碩士班

## 碩 士 論 文

應用機器學習於日文詞彙文字探勘探討商品評價之情

緒分析

Machine Learning in Japanese Text Mining to Explore the  
Sentiment Analysis of Product Evaluation

研 究 生 ： 王祐生

指 導 教 授 ： 林 慶 昌 博 士

中 華 民 國 110 年 6 月 6 日

應用機器學習於日文詞彙文字探勘探討商品評價之情緒分析  
Machine Learning in Japanese Text Mining to Explore the Sentiment  
Analysis of Product Evaluation

研 究 生：王祐生

Student：You-Sheng Wang

指導教授：林慶昌 博士

Advisor：Dr. Ching-Chang Lin

臺北城市科技大學

資訊管理系電子商務碩士班

碩士論文

A Thesis Submitted to  
Master Program of E-Commerce Department of Information Management  
College of Business and Management  
Taipei City University of Science and Technology  
in Partial Fulfillment of the Requirements  
for the Degree of  
Master  
In  
Information Management  
June 2021  
Taipei, Taiwan

中 華 民 國 110 年 6 月 6 日

# 碩士學位論文考試委員會審定書

本校 資訊管理系電子商務碩士班 王祐生 君

所提論文 應用機器學習於日文詞彙文字探勘探討商品評價之情緒分析

經本委員會審定通過，合於碩士資格，特此證明。

## 學位考試委員會

委 員：

張建弘

張慧芬

林慶昌

指導教授：

林慶昌

主 任：

黃大炎

中 華 民 國 110 年 6 月 4 日

# 應用機器學習於日文詞彙文字探勘探討商品 評價之情緒分析

研究生：王祐生

指導教授：林慶昌 博士

臺北城市科技大學資訊管理系電子商務碩士班

## 摘 要

近年來情緒分析技術發展興盛，許多國家在情緒分析技術方面皆發展出各自的方法，且逐漸朝向以機器學習做處理。本研究欲探討日文情緒分析方法，先設計出基於傳統處理方式且適用於日文評價資料之情緒分析方法，然後與目前以存在之基於機器學習法的日文情緒分析演算法一起比較兩者的準確度，最後結合傳統方法與機器學習法，測試兩者配合運作是否提升情緒分析之準確度。本研究提出兩者的結合可保有情緒詞庫的可擴充性，且可在詞庫擴充前的情況下保有一定的分析準確率，此外亦可減輕人力的負擔。

關鍵字：機器學習、文字探勘、情緒分析

Machine Learning in Japanese Text Mining to Explore the Sentiment  
Analysis of Product Evaluation

Student : You-Sheng Wang

Advisor : Dr. Ching-Chang Lin

Submitted to Master Program of E-Commerce Department of Information  
Management

College of Business and Management

Taipei City University of Science and Technology

ABSTRACT

In recent years, the development of sentiment analysis technology has prospered. Many countries have developed their own methods in sentiment analysis technology, and are gradually turning to machine learning for processing. This research intends to explore Japanese sentiment analysis methods. First, it designs a sentiment analysis method that is based on traditional processing methods and is suitable for Japanese evaluation data, and then compares the accuracy of the two with the existing Japanese sentiment analysis algorithm based on machine learning. , Finally, combine traditional methods and machine learning methods to test whether the two work together to improve the accuracy of sentiment analysis. This study proposes that the combination of the two can maintain the scalability of the emotional vocabulary, and it can maintain a certain analysis accuracy before the vocabulary is expanded. In addition, it can reduce the burden of manpower.

Keywords: Machine Learning, Text Mining, Sentiment Analysis

# 誌 謝

衷心感謝我的指導教授林慶昌博士，從一開始的決定論文題目、訂定研究目標，到終於開始進行研究以及撰寫論文內容，老師都給了我很多的幫助。特別是我當時一直無法決定出論文的題目，更別說訂定出研究的目標，這讓我想起剛進大學的時候，系主任跟我和全班同學講過的一句話：「不是不會做，是不知道做什麼」；之後一個衝動下踏入了碩士階段且什麼也沒多想，然後看著身邊的同學都逐漸對論文有了想法，而自己卻還在原地踏步，這時多虧有老師的指引，我才終於找到了自己的研究方向。感謝老師的指點，在我的論文內容或是研究過程出現問題時提供幫助，也感謝曹修源教授和洪慧芬教授幫助我找出論文中的問題點。

此外我要特別感謝教我日文的老師，如果沒有她教會我日文，我也就不會寫出這篇論文。當然也感謝所有教會我寫程式的老師們，成為我這篇研究的基石。最後感謝班導師、同學們和家人給予我的鼓勵。

王祐生 謹致

於臺北城市科技大學資訊管理系電子商務碩士班

2021 年 6 月

# 目次

摘要 .....	iii
ABSTRACT .....	iv
誌謝 .....	v
目次 .....	vi
表次 .....	viii
圖次 .....	ix
公式 .....	x
程式碼 .....	xi
第一章 緒論 .....	1
1.1 研究背景與動機 .....	1
1.2 研究目的 .....	2
1.3 研究範圍與限制 .....	2
1.4 研究方法與流程 .....	2
1.5 論文架構 .....	5
第二章 文獻探討 .....	6
2.1 自然語言處理 .....	6
2.2 情緒分析 .....	6
2.2.1 情緒字典 .....	8
2.3 機器學習 .....	9
2.3.1 RNN 演算法 .....	9
2.3.2 LSTM 演算法 .....	10
第三章 研究方法 .....	11
3.1 研究架構 .....	11
3.2 資料蒐集與整理 .....	11
3.3 資料前處理 .....	12
3.4 情緒分析 .....	13
3.4.1 情緒詞彙之收集 .....	14
3.4.2 情緒詞庫之建立 .....	14
3.4.3 擴充詞庫 .....	17
3.4.4 情緒分析方法 .....	18
3.4.5 機器學習情緒分析方法 .....	23
3.4.6 與機器學習法之結合 .....	25
3.5 分析結果之評估 .....	31
第四章 研究結果 .....	32
4.1 各情緒分析方法之分析結果 .....	32
4.2 擴充詞庫前後之分析結果 .....	36

4.3 結合機器學習之分析結果 .....	39
第五章 結論與未來建議.....	43
5.1 結論 .....	43
5.2 未來研究建議 .....	43
參考文獻 .....	45
中文參考文獻.....	45
英文參考文獻.....	45
日文參考文獻.....	46



# 表 次

表 3-1 意見詞庫樣式(1).....	16
表 3-2 意見詞庫樣式(2).....	16
表 3-3 程度詞庫樣式.....	17
表 3-4 表情符號配分表樣式.....	17
表 3-5 擴充詞庫.....	17
表 4-1 情緒分析方法與評價資料使用情況(各情緒分析方法之分析).....	32
表 4-2 情緒分析方法與評價資料使用情況(擴充詞庫前後之分析).....	36
表 4-3 情緒分析方法與評價資料使用情況(結合機器學習之分析).....	39

# 圖次

圖 1-1 研究流程.....	4
圖 3-1 研究架構.....	11
圖 3-2 資料蒐集與整理之流程.....	12
圖 3-3 情緒詞庫建立示意圖.....	15
圖 3-4 情緒詞庫架構.....	15
圖 3-5 情緒分析流程.....	18
圖 4-1 各情緒分析方法之整體分析結果.....	33
圖 4-2 各情緒分析方法之分析結果(J-AFINN 對日本酒).....	33
圖 4-3 各情緒分析方法之分析結果(Asari 對日本酒).....	34
圖 4-4 各情緒分析方法之分析結果(J-AFINN 對啤酒).....	34
圖 4-5 各情緒分析方法之分析結果(Asari 對啤酒).....	35
圖 4-6 擴充詞庫前後之整體分析結果.....	37
圖 4-7 擴充詞庫前後之分析結果(J-AFINN 詞庫擴充前).....	37
圖 4-8 擴充詞庫前後之分析結果(J-AFINN 詞庫擴充後).....	38
圖 4-9 擴充詞庫前後之分析結果(Asari, 無詞庫擴充).....	38
圖 4-10 結合機器學習之整體分析結果.....	39
圖 4-11 結合機器學習之分析結果(日本酒前 100).....	40
圖 4-12 結合機器學習之分析結果(啤酒).....	40
圖 4-13 結合機器學習之分析結果(日本酒後 100).....	41
圖 4-14 結合機器學習之分析結果(日本酒後 100, 調整實驗條件).....	42
圖 4-15 結合機器學習之分析結果(日本酒後 100, 調整實驗條件, 長條圖).....	42

# 公 式

公式 3-1 情趣分析計算公式 (1).....	18
公式 3-2 情趣分析計算公式 (2).....	19

# 程式碼

程式碼 3-1 日文斷詞程序之程式碼.....	13
程式碼 3-2 日文情緒分析程序之程式碼.....	19
程式碼 3-3 機器學習情緒分析程序之程式碼.....	24
程式碼 3-4 結合機器學習之情緒分析程序之程式碼.....	26

# 第一章 緒論

本章節為本研究之概念性描述，內容共分為五小節，第一節為研究背景與動機、第二節為研究目的、第三節為研究範圍與限制、第四節為研究方法與流程、第五節為論文架構。

## 1.1 研究背景與動機

機器學習是人工智慧的其中一個分支，機器學習主要在探討讓電腦自己學習的演算法，該演算法從大量資料中分析取得規律然後對未知的資料進行預測。機器學習目前被廣泛應用於資料探勘和自然語言處理等方面。文字探勘是透過機器學習和資料探勘以及其他技術達成，指從非結構化的訊息中擷取出重要的資訊，由於超過八成的資訊都是以文字訊息的方式儲存，因此具有高度的潛在商業價值。而情緒分析主要是透過文字探勘達成。

情緒分析是一種用於分析發言者在發言時當下的情緒狀態的技術，利用文字探勘從訊息中擷取出主觀的資訊並換算成數值以便計算訊息的情緒值，屬於文本處理技術的一種。情緒分析可應用於各方面的資料分析，目前較常被應用於商業方面的評價資料分析，以協助企業更有效率的處理大量評價資料。

近年來情緒分析技術發展興盛，許多國家在情緒分析技術方面皆發展出各自的方法，且逐漸朝向以機器學習做處理。本研究欲探討日文情緒分析方法，分別以自然語言處理之傳統情緒分析方法和機器學習情緒分析方法做情緒分析。因此本研究需要其他情緒分析演算法的支持，特別是適用於日文或其他亞洲語言的情緒分析演算法。目前除了關於歐美國家語言的研究資料外，與日文同為亞洲語言且架構相似的中文方面也有相關研究資料可參考，例如建立否定詞庫和程度詞庫、使用字詞權重等。

## 1.2 研究目的

基於上述研究動機，本研究欲設計出基於自然語言處理之傳統處理方式且適用於日文評價資料之情緒分析方法，與目前以存在之基於機器學習法的日文情緒分析演算法一起比較兩者的效能，並測試兩者配合運作的可行性。

依據研究目的，劃分以下三點：

1. 建立適用於日文評價資料的自然語言處理傳統情緒分析法
2. 比較自然語言處理傳統情緒分析法與機器學習情緒分析法之分析結果
3. 結合自然語言處理傳統情緒分析法與機器學習情緒分析法並驗證是否提升效能

## 1.3 研究範圍與限制

本研究針對日文評價資料進行情緒分析，所設計之情緒分析方法僅適用於日文本資料，故本研究之研究範圍僅限於日文評價資料。

本研究所使用之評價資料取自日本亞馬遜網站。由於其商品品項眾多，此外所選定之商品種類數量牽涉到情緒分析方法所適用的商品種類範圍，考量本研究僅探討情緒分析之方法，故本研究最終決定僅以酒類商品做情緒分析方法之探討，其中選用日本酒之評價資料是為了作為日本商品之代表，而選用啤酒之評價資料是因為其商品種類範圍與日本酒相同，此外日本酒與啤酒為日本地區兩大酒類商品，故選用此兩種商品，關於情緒分析方法的運作與限制將於文獻探討之章節做說明。

評價資料方面分為日本酒與啤酒之評價資料，分別隨機篩選出 200 筆與 100 筆，篩選出之評價資料皆為 2019 年至 2020 年間的資料，資料筆數方面是根據研究方式做評價資料之分配，關於研究方式與資料分配之詳細內容將於研究方法之章節做說明。

## 1.4 研究方法與流程

藉由文獻探討找出可支持本研究的情緒分析演算法，然後進行實作。首先以網路爬蟲方式抓取日本亞馬遜之商品評價資料，並以日文斷詞工具做資料前處理。其

次，根據找到的情緒分析演算法建立日文情緒分析用之詞庫如意見詞庫和程度詞庫等，以及制定日文情緒分析之情緒值計算方式，以進行日文情緒分析，並與機器學習情緒分析方法做分析結果之比較。圖 1-1 為本研究之研究流程。

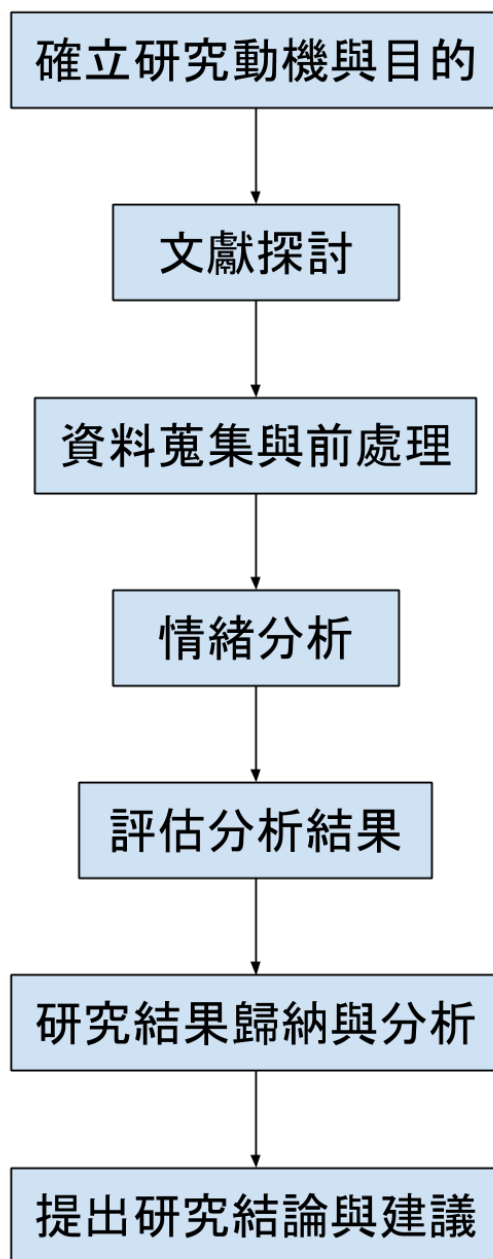


圖 1-1 研究流程



## 1.5 論文架構

本研究共分為五個章節，各章節編排方法如下：

### 第一章 緒論

說明本論文之研究背景與動機、研究目的和研究範圍與限制。

### 第二章 文獻探討

針對自然語言處理、情緒分析和機器學習進行文獻回顧與探討。

### 第三章 研究方法

說明資料搜集、日文斷詞和情緒分析之應用。

### 第四章 研究結果

將實例套用到所提出的研究方法，並將實驗結果進行評估及分析。

### 第五章 結論

總結本研究的貢獻以及後續發展的研究方向。

## 第二章 文獻探討

本章依據相關之文獻，定義並說明既有文獻發展。本章將探討「自然語言處理」、「情緒分析」以及「機器學習」，以作為本文之理論基礎。

### 2.1 自然語言處理

根據 IBM 的統計資料，從 2017 年開始每天都會生成約 2.5 EB 的資料，這些資料中有超過半數都屬於非結構化資料，包括電子郵件訊息和社群媒體文字等。如果把這些資料平均分給全世界所有人處理，則每人每天要處理約 300 MB 的資料。

自然語言處理 (NLP, Natural Language Processing) 是現今社會中作為理解和處理非結構化資料的重要工具之一，其目的是讓機器能夠理解人類說的話或留下的文字訊息 (Thushan, 2019)。情緒分析技術即是利用自然語言處理的方式分析出非結構化資料中的情感訊息。現在許多自然語言處理的任務都使用機器學習的方式處理，因為機器學習在自然語言處理方面已有傑出的表現，且能節省人力。

Thushan(2019) 將自然語言處理的任務分為分析型任務 (Analysis) 與生成型任務 (Generation)，分析型任務中又在細分為句法 (Syntactic)、語義 (Semantic) 和實務 (Pragmatic)。情緒分析技術包含了句法中的分詞任務 (Tokenization) 與語義中的句子 / 摘要分類任務 (Sentence / Synopsis classification)，分詞任務是將文本資料切分成單詞以利電腦處理，句子 / 摘要分類任務則是將文本資料做分類，例如透過情緒分析可將訊息分類為正向與負向，而目前的情緒分析除了能分類訊息的正負向之外，亦可進一步分析出訊息的情感程度（即情緒分數）。

### 2.2 情緒分析

情緒分析屬於文本處理技術的一種，目的是為了分析發言者在發言時當下的情緒狀態，例如發言內容中出現較多正面詞彙如「美麗的」則該發言內容的情緒狀態

偏向正面；若發言內容中出現較多負面詞彙如「醜陋的」則該發言內容的情緒狀態偏向負面。近年來出現許多利用情緒分析從文本資料中抽取出情感訊息的研究，目前較常被應用於商業方面的評價資料分析。竹內広宜與大野正樹（2018）提到網路上存在大量關於商品或服務的相關資訊和評價，因此消費者在購買商品或服務前通常會先在網路上參考這些資訊和評價後再決定是否購買。

游和正、黃挺豪與陳信希（2012）及 Yousef、Medhat 與 Mohamed（2014）皆指出評價資料分析之所以重要是因為產品的評論可讓企業發現產品的優缺點並加以改進。然而林彩雯（2015）認為網路上的評價資料通常是大量的且雜亂無章，若依靠人工處理則失去效率。竹內広宜與大野正樹（2018）認為一個能分析網路上大量評價資料並從中取得消費者喜好與消費者購物行為模式的系統對與企業而言是重要的。而情緒分析技術可協助企業更有效率的處理大量評價資料。

情緒分析有許多方法。Yousef et al. (2014) 將情緒分析大致分為監督式學習法和非監督式學習法。其中監督式學習法常見的方式如字典法和機器學習法。字典法是透過詞庫（情緒字典）對語句做特徵的識別，並配合一套情緒值的計算方式進行情緒分析。而機器學習法之運作方法是在所有訓練資料上加上對應的標籤，然後讓電腦學習將每一筆資料與對應的標籤做關聯（Chollet, 2019）。Thushan (2019) 提出一套情緒分析的傳統做法，在特徵工程（Feature Engineering）的階段中提到一種作為外部資源的詞庫的做法。換言之字典法的情緒分析方法被認為是傳統的情緒分析方法。

在自然語言處理的章節中提到目前的情緒分析技術已能進一步分析出訊息的情感程度。Thelwall et al. (2010) 在研究中提到大多數情緒分析術都試著識別文本資料並分類文本資料的情感極性如正向、負向或中立，儘管這在大多數情況下已經足夠，然而文本資料通常是包含正負情緒的混合，在某些情況下需要同時檢測並表達文本資料的情感程度。因此就如前段所述，情緒分析方法中皆包含有一套情緒值的計算方式。例如 Nielsen (2011) 設計的 AFINN，其計算總情緒值的方式主要是

加總所有文本中情緒詞彙的情緒分數。而林彩雯（2015）則提出以字詞權重的計算方法配合程度詞詞庫與否定詞詞庫以改善中文的情緒分析效能。

### 2.2.1 情緒字典

大部分情緒分析方法之運作仰賴於其情緒字典，其中有一組已預先整理之情緒詞彙清單，包括正面詞彙與負面詞彙（Yousef et al., 2014）。以 Nielsen (2011) 設計的 AFINN 為例，其情緒字典是以一組情緒詞彙清單加上各情緒詞彙的情緒分數所構成，情緒詞彙之情緒分數介於 5~-5 分，正負分數分別表現詞彙之正向或負向。此情緒分析方法與另一套情緒分析軟體 SentiStrength 相似，SentiStrength 與 AFINN 同樣使用帶有情緒分數之情緒字典。而 Thelwall et al. (2010) 對 SentiStrength 所做的研究已證實了其效能。

中文情緒分析方面，林彩雯（2015）將情緒字典細分為意見詞庫、程度詞庫和否定詞庫。與 AFINN 相比多了對程度詞和否定詞的識別，以提高中文語義解析的精確度，陳安怡（2017）在其研究中也使用同樣的方法並已證實了其可行性。日文的文法上也經常使用程度詞和否定詞，其中否定詞的識別在日文語義解析中與在中文語義解析中同樣重要，中山貴樹（2012）提出舉例如「美しい」和「美しくない」，前者是單純的形容詞，而後者屬於前者的否定用法，如果只考慮形容詞而不考慮否定用法將導致分析錯誤，故需建立否定詞庫。在程度詞方面，山内崇資、林佑樹與中野有紀子（2013）提出舉例如「とても楽しい」，「とても」是一程度副詞，作為強調「楽しい」該形容詞的程度之用途，會影響語句之情緒程度，故建立程度詞庫可提高日文情緒分析之效能。

單一詞彙可以具有多種語義，根據評價資料之討論範圍的不同，單詞的語義可能有所不同，游和正等人（2012）發現特定單詞在不同的討論範圍中，其代表的語義會有所不同，且會影響情緒分析的結果。若要精準判定多個不同討論範圍的評價資料，則需針對每一個討論範圍的評價資料個別建立不同的意見詞庫。日文評價資料中也會發生特定單詞在不同的討論範圍中所代表的語義不同的情形，森田晋也

與白井靖人(2018)在其研究中提出同樣的問題並發現以增加詞彙語義之方法可改善情緒分析的精準度。而勝間田昇、山岸直秀、隈裕子與鈴木誠(2019)則是利用特定領域的評價資料建立適用於該領域的意見詞庫以進行情緒分析。

此外 Nielsen(2011) 設計的 AFINN 亦將表情符號納入評分。此做法之理由可推斷為由於表情符號可將情感以非文字的方式簡單地表現出來(木村真有、桂井麻里衣, 2016)。換言之表情符號亦像情緒詞彙一樣帶有正負面之區別及情感強度。

## 2.3 機器學習

傳統程式是由開發者輸入規則讓機器照著規則處理資料然後輸出結果；機器學習並非依靠預先輸入的規則，而是輸入訓練資料和對應的答案經過訓練的過程讓機器自行尋找其中的規則 (Chollet, 2019)。

利用傳統的方式進行自然語言處理的效率不高，最主要的原因存在於其前置作業之中。Thushan(2019) 認為傳統的自然語言處理作法需要耗費時間以及人力資源，特別是特徵工程以及資源的創建方面；機器學習法使用類神經模型，可自行從資料中找尋其特徵而無需人工作業，此外傳統自然語言處理中所需的預處理步驟在機器學習的做法中已不再是必要的。

### 2.3.1 RNN 演算法

RNN 演算法也就是遞迴神經網路 (Recurrent Neural Network)，以一個狀態變數持續擷取序列資料中的特定模式，主要作為處理具順序性之序列資料(如文字資料、股價等)的演算法 (Thushan, 2019)。

人們在讀句子時是逐字處理，也就是透過眼睛的掃視並記憶之前的事物以達到能夠理解句子的含意 (Chollet, 2019)。RNN 演算法即是採用同樣的原理所設計出的簡化版本，利用狀態變數在序列資料中持續更新狀態值並以此預測出下一個狀態值。其中使用到的遞迴連結 (Recurrent Connection) 是關鍵，它讓 RNN 演算法根據先前的狀態值對狀態變數做更新，使 RNN 演算法能根據先前與當前的狀態值進行預測 (Thushan, 2019)。

然而以一個標準 RNN 演算法 (Simple RNN) 而言，由於缺乏長期記憶能力因此並不擅長處理較長的序列資料 (Thushan, 2019)。於是後來就有了 LSTM 演算法的出現。

### 2.3.2 LSTM 演算法

LSTM 演算法也就是長短期記憶網路 (Long Short-Term Memory Network)。Hochreiter 和 Schmidhuber 等人發現標準 RNN 演算法存在梯度消失 (Vanishing Gradients)，也就是因神經網路層數過多導致最終無法訓練的問題，於是在 1997 年開發了 LSTM 演算法，以解決上述問題 (Chollet, 2019)。

人類的記憶機制分為長期記憶和短期記憶，大部分資訊進入大腦後只會停留幾秒然後就被遺忘，屬於短期記憶；但若新資訊與舊資訊高度相關或是該資訊重複進入腦中，該資訊就會永久停留於腦中，屬於長期記憶（施威銘研究室，2019）。LSTM 演算法即是基於此概念開發而成。梯度消失問題阻礙了類神經模型對長期相關訊息的學習，由於 LSTM 演算法解決了梯度消失的問題，因此 LSTM 演算法能有比標準 RNN 演算法更長的記憶能力，且 LSTM 演算法在記憶的保留和拋棄上有更好的控制 (Thushan, 2019)。

LSTM 演算法現已被應用於各種自然語言處理的任務上，且已被證實比起標準 RNN 演算法有較好的表現，特別是針對大量資料的情況下 (Thushan, 2019)。因此目前 LSTM 演算法是最常被應用於社群媒體文字之分析任務的演算法。

## 第三章 研究方法

本章將說明資料搜集、日文斷詞和日文情緒分析之應用，並簡單介紹所使用之技術和工具。

### 3.1 研究架構

本研究之研究架構如圖 3-1 所示。

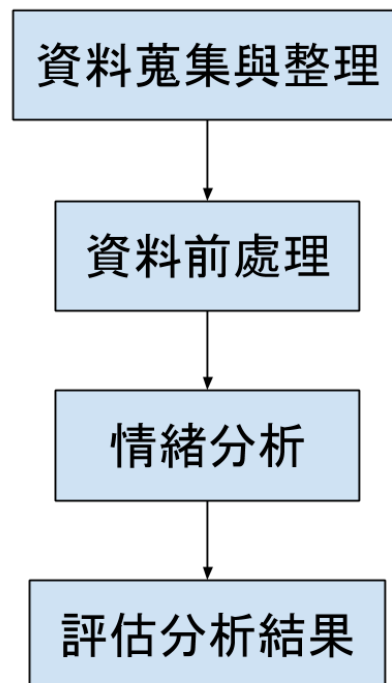


圖 3-1 研究架構

本研究第一步先進行資料蒐集與整理。第二步將資料透過前處理方式處理成電腦可分析的形式，然後進行情緒分析，比較自然語言處理之傳統情緒分析方法與機器學習情緒分析方法之分析結果，最後評估分析結果。

### 3.2 資料蒐集與整理

網路上有大量的評價資料，若依靠人工處理則需花費大量時間。李淑惠(2014)在情緒分析的研究中以網路爬蟲的方式讓程式自動抓取網路上的評價資料。其原

理是利用網路爬蟲工具針對網頁中的 HTML 標籤做分析，然後抓取特定 HTML 標籤內的資料做進一步的使用。

本研究先從網路上抓取評價資料並做整理及製作詞庫，流程如圖 3-2 所示。關於斷詞之詳細內容將於資料前處理之章節做說明。

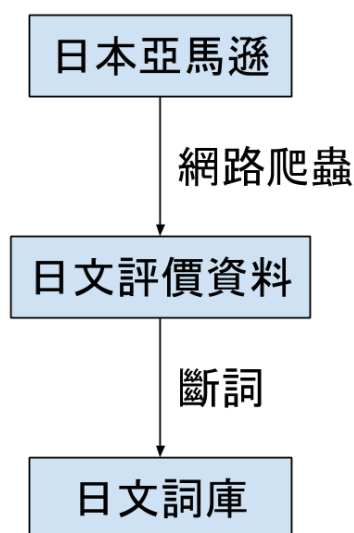


圖 3-2 資料蒐集與整理之流程

本研究所使用之評價資料為日本亞馬遜之商品評價資料，以酒類商品評價資料進行情緒分析之研究。利用網路爬蟲工具分別從日本亞馬遜網站中隨機篩選出日本酒的評價資料 200 筆和啤酒的評價資料 100 筆，共 300 筆評價資料。篩選出之評價資料皆為 2019 年至 2020 年間的資料。

### 3.3 資料前處理

文本資料無法直接透過電腦分析，需要一個前處理方式先將資料處理成單純的字串以便電腦分析 (Hotho, Nürnberger, Paaß, 2005)。本研究以日文評價資料做情緒分析，進行情緒分析前需先對資料做前處理。日文之書寫方式與歐美語言的不同在於歐美語言之書寫方式在詞與詞之間有空格，而日文之書寫方式與中文相同，詞與詞之間無空格，因此需加入斷詞之前處理方式。

本研究使用 Janome 套件對日文評價資料做斷詞之前處理。Janome 是一個自



帶辭典的日文斷詞工具，為打田智子於 2015 年發布之 Python 套件。利用其辭典將日文文本資料切割成個別詞彙，以利後續處理。

本研究在詞庫建立及情緒分析的階段皆使用到斷詞前處理的程序，情緒分析階段中的斷詞前處理程序將於情緒分析之章節做說明，詞庫建立階段中的斷詞前處理程序之程式碼如程式碼 3-1 所示。

程式碼 3-1 日文斷詞程序之程式碼

```
from janome.tokenizer import Tokenizer
t = Tokenizer()

text = "ありがとうございます。"

tokens = t.tokenize(text, wakati = True)
result = ' '.join(tokens)
print(result)
```

Tokenizer 物件用於斷詞處理，tokenize() 方法中的 text 參數為欲做斷詞處理的一段句子，wakati 參數為設定純斷詞模式之用，在一般模式下會連同單詞詞性一併顯示，純斷詞模式則否。此程式碼中先將斷詞結果存放於 tokens 中，再以 join() 方法將斷詞結果合併成單一字串並以空格將各個單詞隔開以便後續處理。

### 3.4 情緒分析

本研究使用自行設計之自然語言處理傳統情緒分析方法與機器學習情緒分析方法分別對篩選出的 100 筆日本酒的評價資料和 100 筆啤酒的評價資料進行情緒分析並比較個自的分析結果。另外針對自行設計之自然語言處理傳統情緒分析方法使用篩選出的另外 100 筆日本酒的評價資料做詞庫擴充前與擴充後的分析結果的比較。最後結合自然語言處理傳統情緒分析方法與機器學習情緒分析方法驗證是否改善分析結果。

本研究自行設計之自然語言處理傳統情緒分析方法以 Nielsen (2011) 設計的 AFINN 為基礎，配合林彩雯 (2015) 提出的字詞權重計算方法運行。AFINN 之運

作方式取自 Nielsen 於 2011 年公開的 AFINN 情緒分析方法，原版是用於英文情緒分析，本研究參考其方法與詞庫改做成日文版本。原版 AFINN 情緒分析方法目前已有適用於 Node.js 和 Python 的套件。

### 3.4.1 情緒詞彙之收集

如同勝間田昇、山岸直秀、隈裕子與鈴木誠（2019）的做法，本研究利用從亞馬遜購物平台收集來的日本酒與啤酒的評價資料，以前述之斷詞前處理程序進一步抽取出適用於日本酒與啤酒的兩組情緒詞彙，整理及建立詞庫並為這些情緒詞彙加入對應的配分。

大多數的情緒詞彙可以直接透過翻譯的方式在 AFINN 的詞庫中找到對應的配分，例如「美味しい」（美味的）的英文翻譯為 delicious，在 AFINN 的詞庫中找到對應的配分為 3。另外也有少數無法直接透過翻譯的方式找到對應配分的詞彙，例如「飲みやすい」無法直接翻譯成單一英文詞彙，在 AFINN 的詞庫中自然也找不到對應的配分，但該詞彙在中文中可解釋成好喝的意思，在日文中與「美味しい」意思相同，因此配分同樣為 3。

日文程度詞的使用方法與中文程度詞的使用方法之間幾乎沒有差異，因此將其轉譯成中文後在林彩雯（2015）所設計的程度詞庫中找出對應的配分即可，例如「とても」在中文裡相當於「很」，參考中文程度詞庫找到對應的配分為 3.5。

Nielsen (2011) 設計的 AFINN 將表情符號納入評分，因此本研究亦將表情符號配分表納入詞庫，配分方面直接比照 AFINN 的配分。

### 3.4.2 情緒詞庫之建立

本研究分別使用篩選出的 100 筆日本酒的評價資料和 100 筆啤酒的評價資料抓取其中的詞彙集成詞庫作為本研究自行設計之自然語言處理傳統情緒分析方法的情緒字典；擴充詞庫方面使用篩選出的另外 100 筆日本酒的評價資料透過同樣的方式對詞庫進行擴充，圖 3-3 為此流程之示意圖。

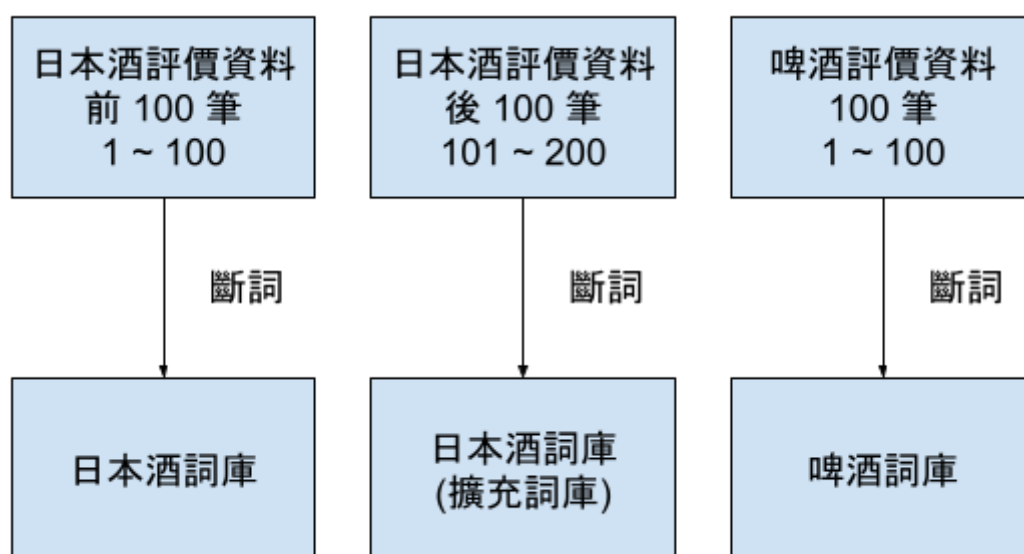


圖 3-3 情緒詞庫建立示意圖

本研究參考 Nielsen (2011) 設計的 AFINN 的情緒詞庫和林彩雯 (2015) 的研究方法將所製作的詞庫分為意見詞庫、程度詞庫和表情符號配分表如圖 3-4 所示。

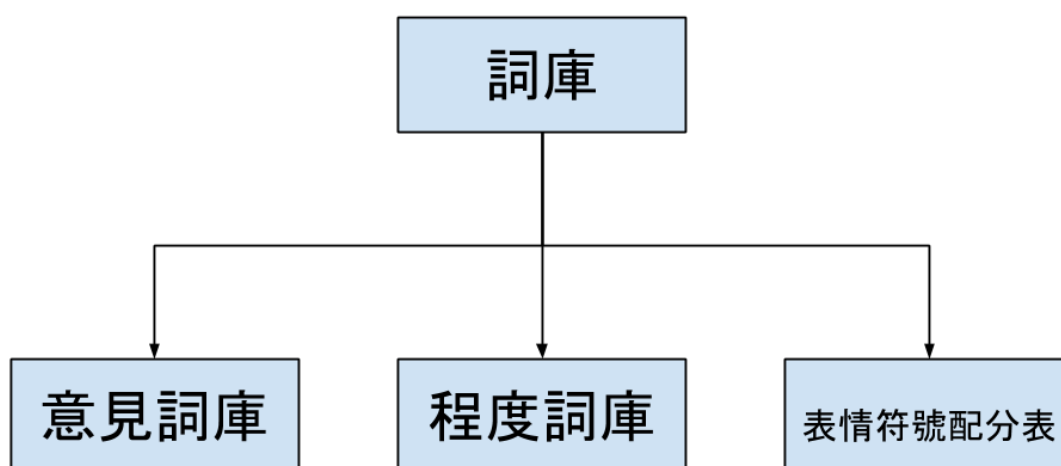


圖 3-4 情緒詞庫架構

意見詞庫內包含正面詞彙和負面詞彙，各情緒詞彙之情緒值參考 Nielsen (2011) 對情緒詞彙的評分標準做設定。之所以省略否定詞庫是因為日文之否定表示方法與中文之否定表示方法有所差距，因此本研究中將否定詞合併於意見詞庫中。

整理出來的詞庫為一表格形式如表 3-1，並做成 csv 格式供程式使用。下表

為意見詞庫之樣式，class 的欄位表示該意見詞適用的評價資料種類如 sake（日本酒）和 beer（啤酒）。main 即為意見詞，詞庫以關鍵詞做表示，分為第 1 關鍵詞 (main) 與第 2 關鍵詞 (sub)，to\_sub 為第 1 關鍵詞到第 2 關鍵詞之間的詞數，如表格第 2 項：ありがとう (main) 之後的第 1(to\_sub) 個詞為ござい (sub)，之所以使用此設計方式的理由將於表 3-2 做進一步的解釋。score 即為該意見詞的情緒分數，程式將以此作為計分標準。

表 3-1 意見詞庫樣式(1)

class	main	to_sub	sub	score
sake	甘い	-	-	2
sake	ありがとう	1	ござい	7
sake	いい	-	-	3
beer	味	3	ない	-3
beer	甘	-	-	2

詞庫以關鍵詞做表示，並分為第 1 關鍵詞與第 2 關鍵詞以解決日文的否定表示及特定意見詞的組成問題如表 3-2 所示：

表 3-2 意見詞庫樣式(2)

class	main	to_sub	sub	score
sake	ありがとう	1	ござい	7
sake	超	1	人気	15
sake	悪く	1	ない	3

表格第 1 項為日文特有的文法，表非常感謝之意，帶有程度上的差異，經斷詞後兩詞分開表示，故採用次作法。該詞庫除了合併有否定詞的表示外，也包含有特定程度詞與名詞的合併用法，表格第 2 項即為此例，由於名詞本身不帶情緒成分（即無法配分），必須與程度詞合併使用才能構成意見詞並給予配分。表格第 3 項即為日文的否定表示，表示不差的意思。

表 3-3 程度詞庫樣式

scan	main	to_sub	sub	score
-2	あまりに	-1	も	5
-1	とても	-	-	3.5
-1	一番	-	-	4.5

表 3-3 為程度詞庫之樣式，與意見詞庫不同的是程度詞不分類別，不管在哪一種評價資料裡表現出來都一樣，因此沒有 class 的欄位。和意見詞庫一樣以關鍵詞做表示，並分為第 1 關鍵詞與第 2 關鍵詞以解決日文文法上的問題。此外多了 scan 的欄位，表示程度詞到意見詞之間的詞數，如表格第 1 項的程度詞 (main) 會出現在意見詞的前第 2(scan) 個詞的位置。

表 3-4 表情符號配分表樣式


main	score
	3
	2

表 3-4 為表情符號配分表之樣式，與意見詞庫類似但結構簡單，和程度詞一樣不分類別且無需考慮文法問題，只針對表情符號做識別並給予配分。

### 3.4.3 擴充詞庫

表 3-5 擴充詞庫

type	class	main	to_sub	sub	score
train	sake	甘い	-	-	2
train	sake	いい	-	-	3
test	sake	安易	-	-	1
test	sake	旨し	-	-	3
test	sake	感謝	-	-	2

表 3-5 說明擴充詞庫的作法，以意見詞庫為例。type 欄位表示詞彙的型態為原始或擴充，train 代表原始詞彙，test 代表擴充詞彙。在情緒分析的程序中，程式會根據 type 欄位是 train 或是 test 將兩種詞彙分開，詞庫擴充前的實驗階段只使用原始詞彙，詞庫擴充後的實驗階段則兩者皆使用。

#### 3.4.4 情緒分析方法

本研究以 AFINN 為基礎，配合字詞權重計算方法以及本研究所製作之日文情緒詞庫，設計出適用於日文評價資料的情緒分析方法，其運作流程如圖 3-5 所示。

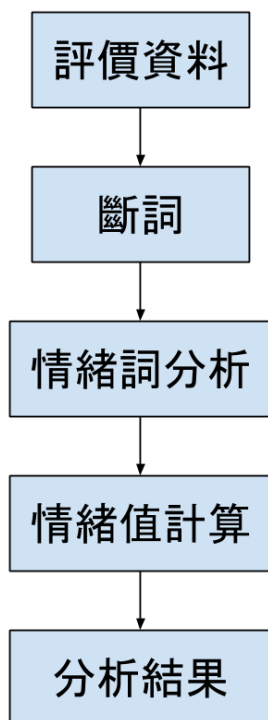


圖 3-5 情緒分析流程

情緒分數的計算方式使用 AFINN 的計算方式，其計算方式為加總所有意見詞和表情符號的配分 (score) 得到情緒分數 (total)，計算公式如公式 3-1 所示。

$$\Sigma total = score_1 + score_2 + \dots + score_n$$

公式 3-1 情緒分數計算公式 (1)

配合字詞權重的計算方法，其計算方式為計算意見詞的配分 (score) 與程度詞的配分 (degree) 之乘積，最終的計算公式如公式 3-2 所示。

$$\Sigma total = (score \times degree)_1 + \dots + (score \times degree)_n$$

公式 3-2 情緒分數計算公式 (2)

此情緒分析階段之程式碼如程式碼 3-2 所示。

程式碼 3-2 日文情緒分析程序之程式碼

```
import csv
from janome.tokenizer import Tokenizer
t = Tokenizer()

step = '4-1'
t_class = 'sake'

final_result = []
result_list = []

data_list = []
data_list_sl = []
score_list = []
score_list_sl_1 = []
score_list_sl_2 = []
degree_list = []
emoticon_list = []
if t_class == 'sake':
    with open('評價資料 - sake.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            data_list.append(row)
            if step == '4-1':
                for dls in range(1, len(data_list)):
                    if data_list[dls][0] == 'train':
                        data_list_sl.append(data_list[dls])
            else:
                for dls in range(1, len(data_list)):
```

```

        if data_list[dls][0] == 'test':
            data_list_sl.append(data_list[dls])
with open('情緒詞.csv', newline = '') as csvfile:
    rows = csv.reader(csvfile)
    for row in rows:
        score_list.append(row)
    for sls in range(1, len(score_list)):
        if score_list[sls][1] == 'sake':
            score_list_sl_1.append(score_list[sls])
if step == '4-1' or step == '4-2bf':
    for sls in range(0, len(score_list_sl_1)):
        if score_list_sl_1[sls][0] == 'train':
            score_list_sl_2.append(score_list_sl_1[sls])
else:
    for sls in range(0, len(score_list_sl_1)):
        score_list_sl_2.append(score_list_sl_1[sls])
elif t_class == 'beer':
    with open('評價資料 - beer.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            data_list.append(row)
        for dls in range(1, len(data_list)):
            data_list_sl.append(data_list[dls])
    with open('情緒詞.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            score_list.append(row)
        for sls in range(1, len(score_list)):
            if score_list[sls][1] == 'beer':
                score_list_sl_2.append(score_list[sls])
with open('程度詞.csv', newline = '') as csvfile:
    rows = csv.reader(csvfile)
    for row in rows:
        degree_list.append(row)
with open('表情符號.csv', newline = '') as csvfile:
    rows = csv.reader(csvfile)
    for row in rows:
        emoticon_list.append(row)

```



```

for dls in range(0, len(data_list_sl)):
    tokens = t.tokenize(data_list_sl[dls][1], wakati = True)
    result = []
    for token in tokens:
        result.append(token)
    total = 0
    final_result = []
    for search in range(len(result)):
        found = ""
        for sls in range(0, len(score_list_sl_2)):
            if result[search] == score_list_sl_2[sls][2]:
                if score_list_sl_2[sls][2] != found:
                    if score_list_sl_2[sls][3] == "-":
                        degree_search = 0
                        for dgls in range(1, len(degree_list)):
                            try:
                                if result[search + int(degree_list[dgls]
[1])] == degree_list[dgls][2]:
                                    degree_search = 1
                                    if degree_list[dgls][3] == "-":
                                        total += (float(score_list_sl_2
[sls][5]) * float(degree_list[dgls][5]))
                                    break
                            except:
                                continue
                        if degree_search == 0:
                            total += float(score_list_sl_2[sls][5])
                        else:
                            try:
                                if result[search + int(score_list_sl_2[sls]
[3])] == score_list_sl_2[sls][4]:

```

```

        found = score_list_sl_2[sls][2]
        degree_search = 0
        for dgls in range(1, len(degree_list)):
            try:
                if result[search + int(degree_list[dgls][1])] == degree_list[dgls][2]:
                    degree_search = 1
                    if degree_list[dgls][3] == "-":
                        total += (float(score_list_sl_2[sls][5]) * float(degree_list[dgls][5]))
                        break
                    else:
                        if result[search + int(degree_list[dgls][3])] == degree_list[dgls][4]:
                            total += (float(score_list_sl_2[sls][5]) * float(degree_list[dgls][5]))
                            break
                        except:
                            continue
            if degree_search == 0:
                total += float(score_list_sl_2[sls][5])
            except:
                continue
        else:
            continue
        for els in range(1, len(emoticon_list)):
            if result[search] == emoticon_list[els][1]:
                total += int(emoticon_list[els][2])
        if total > 0:
            final_result.append('positive')
        elif total < 0:
            final_result.append('negative')
        else:
            final_result.append('none')
        final_result.append(total)
        result_list.append(final_result)

```

```
with open('result_j-afinn.csv', 'w', newline = '') as csvfile:
    writer = csv.writer(csvfile)
    for rls in range(len(result_list)):
        writer.writerow(result_list[rls])
```

評價資料及情緒詞庫皆以 csv 格式存放，因此該程式先以 csv 套件將評價資料及情緒詞庫讀取並存入個別的 list 清單中。本研究分為三個實驗階段並使用日本酒和啤酒兩組評價資料，此程式碼適用於前兩個實驗階段，利用 step 參數控制欲進行的實驗階段如 4-1（各情緒分析方法之分析）、4-2bf（擴充詞庫前之分析）和 4-2af（擴充詞庫後之分析），利用 t\_class 參數控制欲使用的評價資料如 sake（日本酒）和 beer（啤酒）。

接著利用 Janome 套件做斷詞前處理，先從評價資料清單 (data\_list\_sl) 中讀取一筆評價資料，然後如同前述之斷詞前處理程序以 tokenize() 方法做斷詞處理並將斷詞結果存放於 result 清單中。

最後進行情緒分析之程序，掃描 result 清單內的所有單字，若掃描到的單字亦出現在情緒詞庫 (score\_list\_sl\_2) 中則進入掃描程度詞的階段，在掃描程度詞的階段中會偵測情緒詞的前後是否有出現在程度詞庫 (degree\_list) 中的單字，如果有則計入情緒詞分數乘上程度詞分數之分數，沒有則只計入情緒詞分數。掃描表情符號的階段與掃描情緒詞的階段大致相同，使用表情符號配分表 (emoticon\_list) 但不偵測程度詞，直接計入表情符號之分數。全部掃描完成後即統計總分 (total)，分析結果包含正負向標籤如 positive（正向）、negative（負向）和 none（0分），以及總分 (total)，同樣以 csv 格式存放。

### 3.4.5 機器學習情緒分析方法

本研究利用目前已存在之機器學習情緒分析方法進行情緒分析作為與自行設計之自然語言處理傳統情緒分析方法的對比，最後結合兩者做改善分析結果之測試。

本研究使用 Asari 套件對日文評價資料做情緒分析。Asari 是一個基於機器學習法的日文情緒分析工具，為 Hiroki Nakayama 於 2019 年發布之 Python 套件。透過其類神經模型計算出評價資料的情緒分數。

此情緒分析階段之程式碼如程式碼 3-3 所示。

程式碼 3-3 機器學習情緒分析程序之程式碼

```
import csv
from asari.api import Sonar
sonar = Sonar()

step = '4-1'
t_class = 'sake'

data_list = []
data_list_sl = []
if t_class == 'sake':
    with open('評價資料 - sake.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            data_list.append(row)
            if step == '4-1':
                for dls in range(1, len(data_list)):
                    if data_list[dls][0] == 'train':
                        data_list_sl.append(data_list[dls])
            else:
                for dls in range(1, len(data_list)):
                    if data_list[dls][0] == 'test':
                        data_list_sl.append(data_list[dls])
elif t_class == 'beer':
    with open('評價資料 - beer.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            data_list.append(row)
            for dls in range(1, len(data_list)):
                data_list_sl.append(data_list[dls])

with open('result_asari.csv', 'w', newline = '') as csvfile:
```

```

writer = csv.writer(csvfile)
for dls in range(0, len(data_list_sl)):
    final_result = []
    result_list = []
    output = sonar.ping(text = data_list_sl[dls][1])
    result_1 = output['top_class']
    if output['top_class'] == 'negative':
        result_2 = output['classes'][0]['confidence']
    else:
        result_2 = output['classes'][1]['confidence']
    final_result.append(result_1)
    final_result.append(result_2)
    result_list.append(final_result)
    writer.writerow(result_list[0])

```

同樣先以 csv 套件將評價資料讀取並存入個別的 list 清單中，但不使用情緒詞庫。此程式碼亦適用於前兩個實驗階段，利用 step 參數控制欲進行的實驗階段，利用 t\_class 參數控制欲使用的評價資料。

先從評價資料清單 (data\_list\_sl) 中讀取一筆評價資料，然後使用 sonar.ping() 方法進行情緒分析，分析結果為 dict 型態，包括 top\_class(正負向標籤)和 classes (情緒分數)，情緒分數又分負向分數 ([‘classes’][0][‘confidence’]) 和正向分數 ([‘classes’][1][‘confidence’])，取出正負向標籤和情緒分數後以 csv 格式存放。

### 3.4.6 與機器學習法之結合

本研究將探討自然語言處理之傳統情緒分析方法結合機器學習演算法是否可提高分析準確率，當本研究自行設計之自然語言處理傳統情緒分析方法的分析結果與機器學習情緒分析方法的分析結果不一致時則使用機器學習情緒分析方法的分析結果，並乘上兩者之間的平均差距（本研究之計算結果為：7.524 分）使其接近本研究自行設計之自然語言處理傳統情緒分析方法的評價分數。

由於套件相容性的問題，兩種情緒分析方法無法同時運作，因此此情緒分析階段為前兩個情緒分析階段分析結果的模擬操作。此情緒分析階段之程式碼如程式

碼 3-4 所示。

程式碼 3-4 結合機器學習之情緒分析程序之程式碼

```
import csv

step = 1
t_class = 'sake'
care_zero = False

result_list = []
result_list_sl = []
result_list_jafinn = []
result_list_asari = []

if t_class == 'sake':
    with open('評價資料 - sake.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            result_list.append(row)
            if step == 1:
                for dls in range(1, len(result_list)):
                    if result_list[dls][0] == 'train':
                        result_list_sl.append(result_list[dls])
            else:
                for dls in range(1, len(result_list)):
                    if result_list[dls][0] == 'train':
                        result_list_sl.append(result_list[dls])
else:
    with open('評價資料 - beer.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            result_list.append(row)
            for dls in range(1, len(result_list)):
                result_list_sl.append(result_list[dls])

if t_class == 'sake':
    if step == 1:
        with open('result_j-afinn_s41.csv', newline = '') as csvfile:
```

```

        rows = csv.reader(csvfile)
        for row in rows:
            result_list_jafinn.append(row)
    else:
        with open('result_j-afinn_s42bf.csv', newline = '') as csvfile:
            rows = csv.reader(csvfile)
            for row in rows:
                result_list_jafinn.append(row)
else:
    with open('result_j-afinn_b41.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            result_list_jafinn.append(row)

if t_class == 'sake':
    if step == 1:
        with open('result_asari_s41.csv', newline = '') as csvfile:
            rows = csv.reader(csvfile)
            for row in rows:
                result_list_asari.append(row)
    else:
        with open('result_asari_s42.csv', newline = '') as csvfile:
            rows = csv.reader(csvfile)
            for row in rows:
                result_list_asari.append(row)
else:
    with open('result_asari_b41.csv', newline = '') as csvfile:
        rows = csv.reader(csvfile)
        for row in rows:
            result_list_asari.append(row)

file = open('index.html', 'w')
file.write("<!DOCTYPE html>\n")
file.write("<script src='https://code.jquery.com/jquery-3.5.1.min.js'></script>\n")
file.write("<script src='https://code.highcharts.com/highcharts.js'></script>\n")
file.write("<html>\n")

```

```

file.write("<head>\n")
file.write("<title>Test</title>\n")
file.write("</head>\n")
file.write("<body>\n")
file.write("<div id='container' style='width:100%; height:100%;'>\n")
file.write("<script>\n")
file.write("var data = {\n")
file.write("chart: {type: 'line'},\n")
file.write("title: {text: '情緒分析結果'},\n")
file.write("xAxis: {categories: [")

for i in range(1, 100):
    file.write("'" + str(i) + "', ")
file.write("'100'")

file.write("]},\n")
file.write("yAxis: {title: {text: '情緒分數'}},\n")
file.write("plotOptions: {line: {dataLabels: {enabled: false}, enableMouseTracking: false}},\n")
file.write("series: [{\n")
file.write("name: '專家評分', data: [")

for i in range(len(result_list_sl) - 1):
    file.write(str(result_list_sl[i][2]) + ", ")
file.write(result_list_sl[99][2])

file.write("]}, {\n")
file.write("name: 'J-AFINN + Asari', data: [")

for i in range(len(result_list_jafinn) - 1):
    if not care_zero:
        if float(result_list_jafinn[i][1]) > 0.0 and result_list_asari[i][0] == "positive":
            file.write(str(result_list_jafinn[i][1]) + ", ")
        elif float(result_list_jafinn[i][1]) < 0.0 and result_list_asari[i][0] == "negative":
            file.write(str(result_list_jafinn[i][1]) + ", ")
        elif float(result_list_jafinn[i][1]) == 0.0:

```



```

        file.write(str(result_list_jafinn[i][1]) + ", ")
    else:
        if result_list_asari[i][0] == "positive":
            file.write(str(float(result_list_asari[i][1]) * 7.524)
+ ", ")
        else:
            file.write(str(float(result_list_asari[i][1]) * -
7.524) + ", ")
    else:
        if float(result_list_jafinn[i][1]) > 0.0 and result_list_asari[
i][0] == "positive":
            file.write(str(result_list_jafinn[i][1]) + ", ")
        elif float(result_list_jafinn[i][1]) < 0.0 and result_list_asar
i[i][0] == "negative":
            file.write(str(result_list_jafinn[i][1]) + ", ")
        else:
            if result_list_asari[i][0] == "positive":
                file.write(str(float(result_list_asari[i][1]) * 7.524)
+ ", ")
            else:
                file.write(str(float(result_list_asari[i][1]) * -
7.524) + ", ")
if not care_zero:
    if float(result_list_jafinn[99][1]) > 0.0 and result_list_asari[99]
[0] == "positive":
        file.write(str(result_list_jafinn[99][1]))
    elif float(result_list_jafinn[99][1]) < 0.0 and result_list_asari[9
9][0] == "negative":
        file.write(str(result_list_jafinn[99][1]))
    elif float(result_list_jafinn[99][1]) == 0.0:
        file.write(str(result_list_jafinn[99][1]))
    else:
        if result_list_asari[99][0] == "positive":
            file.write(str(float(result_list_asari[99][1]) * 7.524))
        else:
            file.write(str(float(result_list_asari[99][1]) * -7.524))
else:

```

```

        if float(result_list_jafinn[99][1]) > 0.0 and result_list_asari[99][0] == "positive":
            file.write(str(result_list_jafinn[99][1]))
        elif float(result_list_jafinn[99][1]) < 0.0 and result_list_asari[99][0] == "negative":
            file.write(str(result_list_jafinn[99][1]))
        else:
            if result_list_asari[99][0] == "positive":
                file.write(str(float(result_list_asari[99][1]) * 7.524))
            else:
                file.write(str(float(result_list_asari[99][1]) * -7.524))

file.write("]]]\n")
file.write("}\n")
file.write("$('#container').highcharts(data);\n")
file.write("</script>\n")
file.write("</div>\n")
file.write("</body>\n")
file.write("</html>\n")
file.close

```

首先讀取前兩個情緒分析階段的分析結果並存入個別的 list 清單中，利用 step 參數與 t\_class 參數控制欲使用的分析結果資料，然後模擬兩種情緒分析方法結合的分析結果。

當兩者的分析結果一致時使用本研究自行設計之自然語言處理傳統情緒分析方法的分析結果，不一致時則使用機器學習情緒分析方法的分析結果並乘上兩者之間的平均差距（7.524 分）使其接近本研究自行設計之自然語言處理傳統情緒分析方法的評價分數。

本研究自行設計之自然語言處理傳統情緒分析方法的分析結果可能會出現 0 分的分析結果，若將此分析結果考慮進去就會影響最終模擬分析的結果，透過 care\_zero 參數可控制是否考慮本研究自行設計之自然語言處理傳統情緒分析方法的分析結果為 0 分的分析結果，預設情況下設定為否。

### 3.5 分析結果之評估

本研究先請教日文方面之專家檢查評價資料之原文文意並對評價資料之原始資料訂出標準評分，接著將評價分析所分析出之評分結果與標準評分作對照，比較分析結果與標準評分之接近程度，做效能之驗證。

本研究請教楊姓日文學者做評價資料的原文文意檢查。參考本研究設計之詞庫並以人工方式計算標準評分。

## 第四章 研究結果

以下是情緒分析的分析結果，利用簡單的折線圖顯示分析結果與專家評分兩折線之間的相似度，進一步得知該情緒分析的準確率。「J-AFINN」為本研究所設計之傳統情緒分析方法的代稱，「Asari」為基於機器學習法之日文情緒分析工具的名稱，「專家評分」為本研究所請教之楊姓日文學者的代稱。首先是各情緒分析的分析結果，接著是針對 J-AFINN 做詞庫擴充前後的分析結果。所選用之商品名稱如下。

日本酒：獺祭(だっさい) 純米大吟釀45 1800ml，共篩選出 200 筆資料。

啤酒：【新ジャンル/第3のビール】アサヒ ザ・リッチ [ ビール 350ml×24本 ]，共篩選出 100 筆資料。

### 4.1 各情緒分析方法之分析結果

首先針對 J-AFINN 和 Asari 做情緒分析結果之評估，分別使用日本酒前 100 筆評價資料和啤酒評價資料 100 筆進行情緒分析，分析出之個別分析結果以折線圖表示，顯示出整體準確率、正負向方面正確率和不考慮中間分數之正負向方面正確率（不考慮 0 分），其中整體準確率為比較兩折線圖相似度所計算出之數值。所使用的情緒分析方法和評價資料如表 4-1 所示，整體分析結果如圖 4-1 所示，以下針對個別分析結果作解析。

表 4-1 情緒分析方法與評價資料使用情況(各情緒分析方法之分析)

		對照 1	對照 2
日本酒	情緒分析方法	J-AFINN	Asari
	評價資料	日本酒前 100 筆評價資料 (1 ~ 100)	
啤酒	情緒分析方法	J-AFINN	Asari
	評價資料	啤酒評價資料 100 筆 (1 ~ 100)	

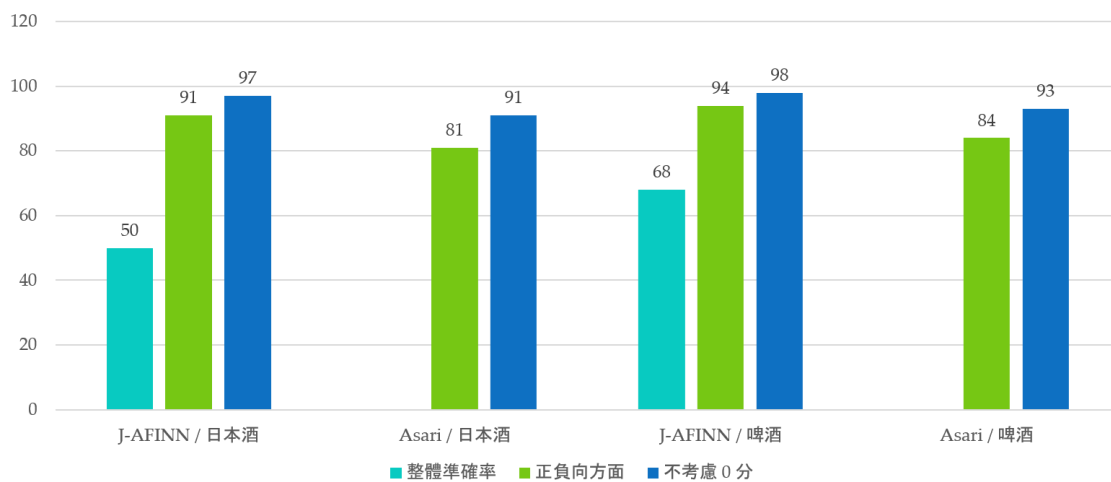


圖 4-1 各情緒分析方法之整體分析結果

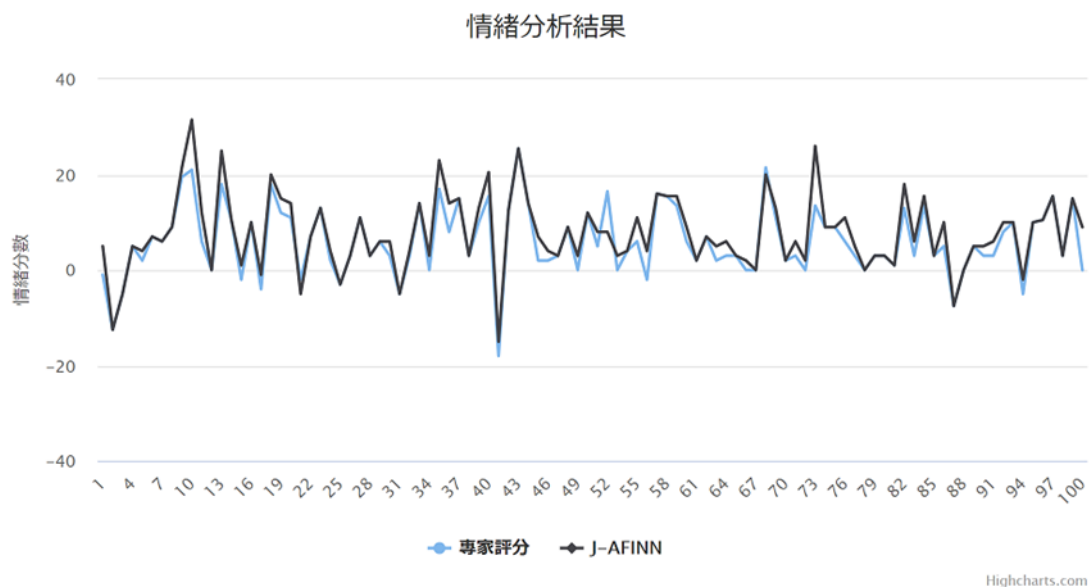


圖 4-2 各情緒分析方法之分析結果(J-AFINN 對日本酒)

圖 4-2 是 J-AFINN 對日本酒前 100 筆評價資料的分析結果，整體準確率（兩折線圖相似度）約有 50%，就正負向方面的正確率約有 91%，而就正負向方面且不考慮中間分數（不考慮 0 分）的正確率約有 97%。從折線圖可以看出兩者之間是相似的，但不完全準確。

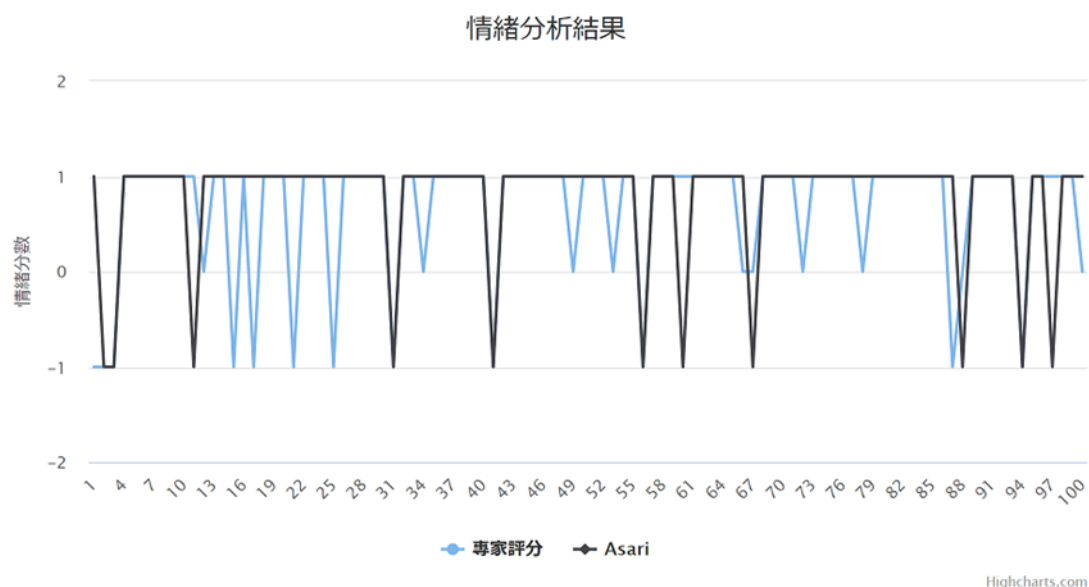


圖 4-3 各情緒分析方法之分析結果(Asari 對日本酒)

圖 4-3 是 Asari 對日本酒前 100 筆評價資料的分析結果，由於採用機器學習的分析方法，其計算方式特殊且不明，就情緒分數方面無法與 J-AFINN 做比較。因此僅考慮正負向的正確率約有 81%，不考慮中間分數的正確率約有 91%。

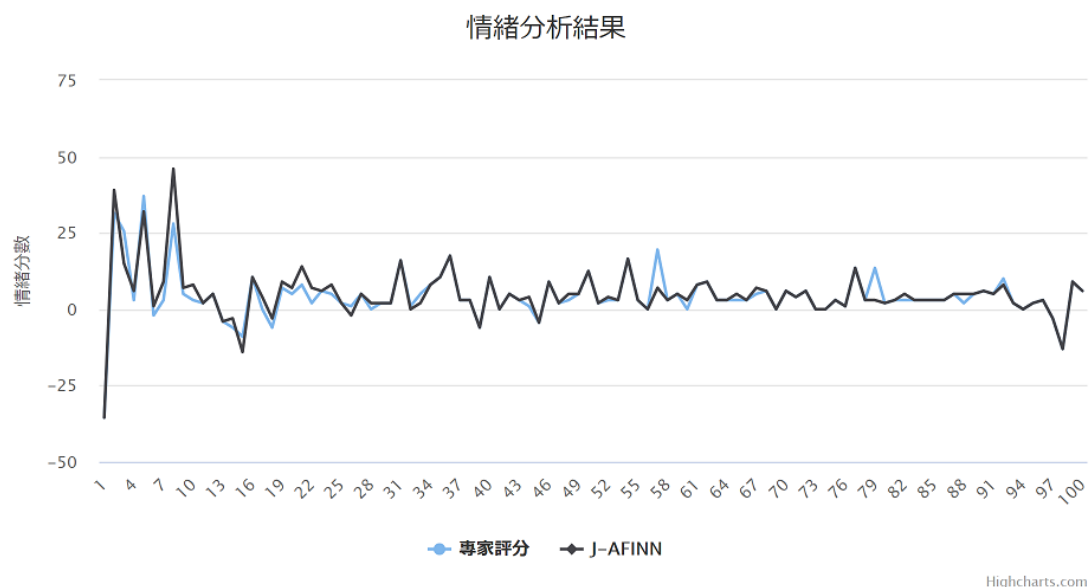


圖 4-4 各情緒分析方法之分析結果(J-AFINN 對啤酒)

圖 4-4 是 J-AFINN 對啤酒評價資料的分析結果，整體準確率約有 68%，正負向方面的正確率約有 94%，正負向方面且不考慮中間分數的正確率約有 98%。

與 J-AFINN 對日本酒評價資料的分析結果相比有較高的準確率。

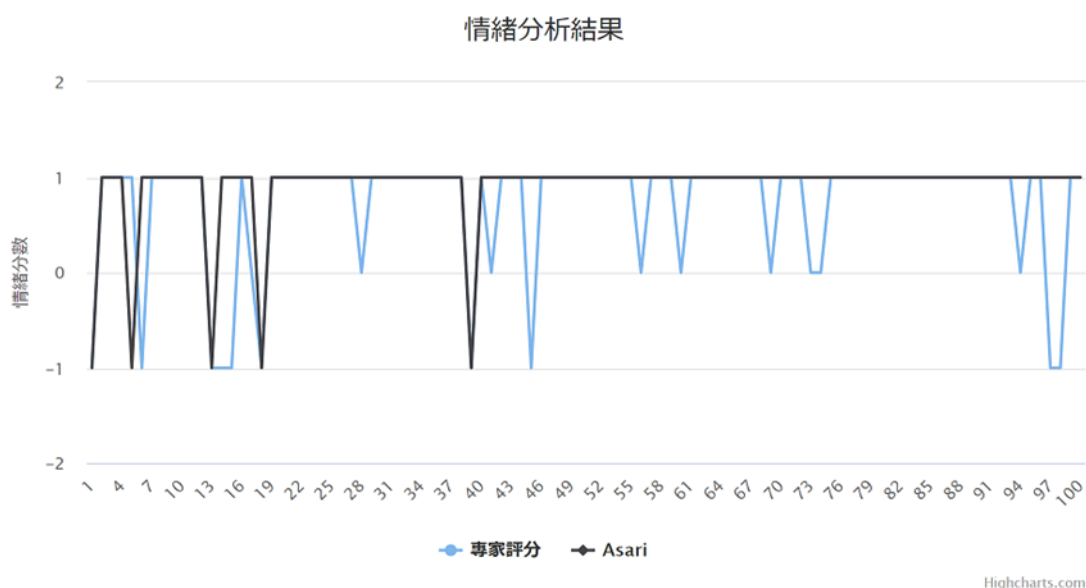


圖 4-5 各情緒分析方法之分析結果(Asari 對啤酒)

圖 4-5 是 Asari 對啤酒評價資料的分析結果，正負向方面的正確率約有 84%，正負向方面且不考慮中間分數的正確率約有 93%。與 Asari 對日本酒評價資料的分析結果相比，正確率稍微高了一點。

從以上分析結果可以看出不管是 J-AFINN 還是 Asari，針對日本酒評價資料的分析準確率都比針對啤酒評價資料的分析準確率低，可以合理推斷問題出在評價資料本身，主要問題出在 J-AFINN 和 Asari 都無法判斷評價資料中情緒詞彙針對的對象，這也是目前現有的情緒分析工具都還難以做到的功能。謝梁、魯穎與勞虹嵐(2018)對情緒分析提出了三個難點，其中一點認為評價資料中的文字特徵難以識別，個別文字討論的主題可能是不同的人事物。日本酒的評價資料中又時常出現情緒詞彙並非針對商品本身的問題，因此導致程式誤判，進而影響分析結果。

然而 Asari 的整體準確率並沒有比 J-AFINN 高，這方面透過文獻探討可以歸納出兩個主要問題。Asari 是基於機器學習法的情緒分析方法，與基於傳統處理方式的情緒分析方法最大的差別在於不需要詞庫，取而代之的是採用類神經模型，由於沒有詞庫所以無法針對不同商品種類做個別的優化。類神經模型需要經過訓

練之後才能使用，根據所使用的訓練資料會影響後續的分析結果。從分析結果可以看出 Asari 對兩種評價資料的準確率差異不大，可以推斷 Asari 這套情緒分析工具使用的訓練資料有足夠的廣泛度，以至於針對兩種評價資料皆有一定的準確率，但也因為使用的訓練資料較為廣泛進而導致整體的準確率不高。

## 4.2 擴充詞庫前後之分析結果

Nielsen (2011) 在研究中證實對詞庫做擴充可提高情緒分析的準確率，因此本研究也針對 J-AFINN 的詞庫擴充前後做比較。此外機器學習的類神經模型在經過訓練之後即可對任何資料進行分析且有一定的準確率，沒有詞庫擴充的問題，因此也會以 Asari 的分析結果做對照。此處研究以日本酒評價資料進行情緒分析，使用日本酒後 100 筆評價資料。所使用的情緒分析方法和評價資料如表 4-2 所示，整體分析結果如圖 4-6 所示，以下針對個別分析結果作解析。

表 4-2 情緒分析方法與評價資料使用情況(擴充詞庫前後之分析)

		詞庫擴充前	詞庫擴充後
對照 1	情緒分析方法	J-AFINN (未擴充詞庫)	J-AFINN (已擴充詞庫)
	評價資料	日本酒後 100 筆評價資料 (101 ~ 200)	
		(無詞庫擴充)	
對照 2	情緒分析方法	Asari	
	評價資料	日本酒後 100 筆評價資料 (101 ~ 200)	



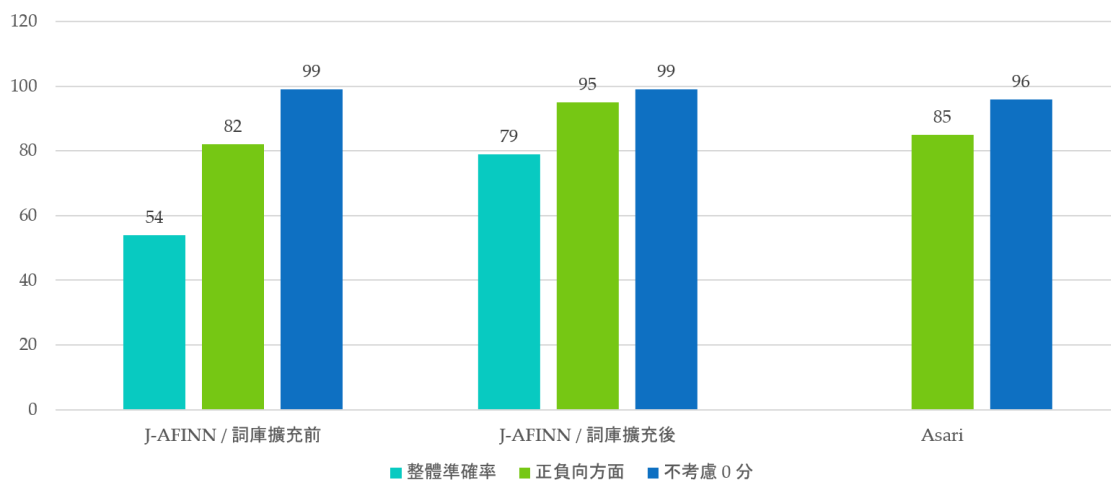


圖 4-6 擴充詞庫前後之整體分析結果

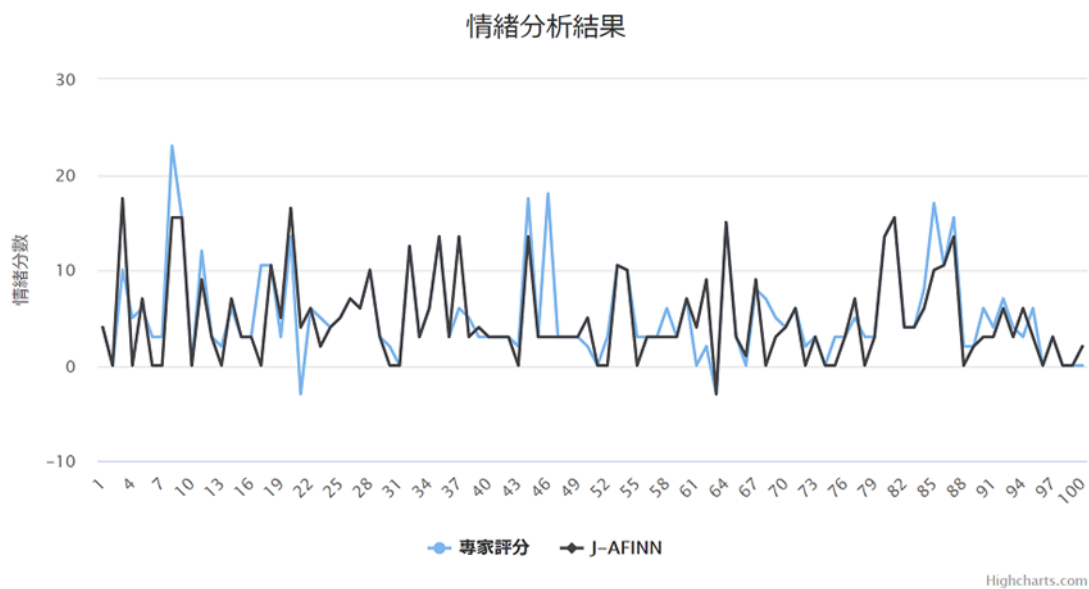


圖 4-7 擴充詞庫前後之分析結果(J-AFINN 詞庫擴充前)

圖 4-7 是 J-AFINN 在詞庫擴充前的分析結果，整體準確率約有 54%，正負向方面的正確率約有 82%，正負向方面且不考慮中間分數的正確率約有 99%。

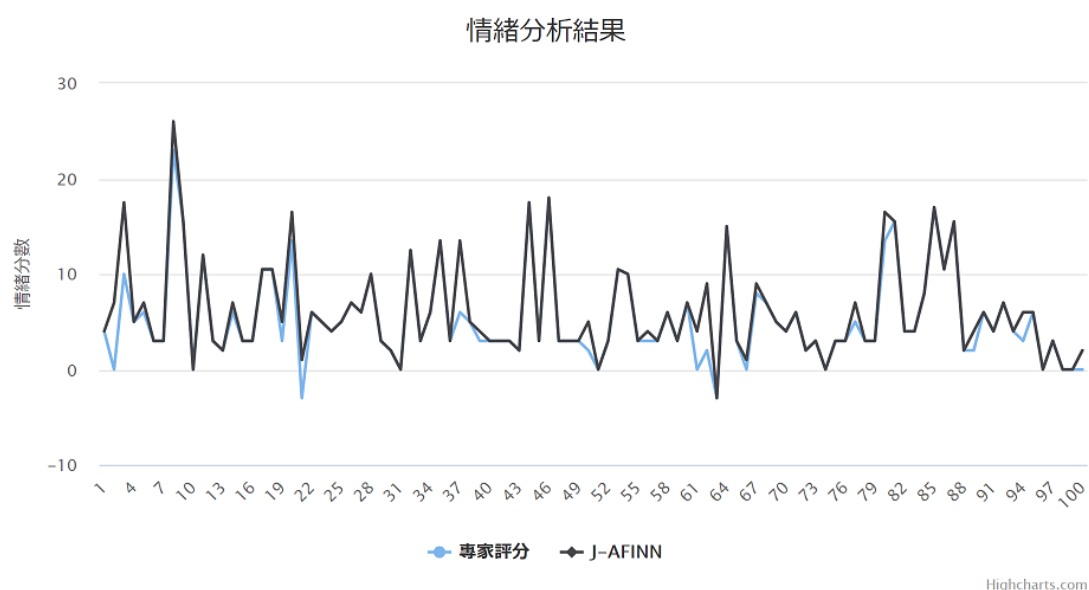


圖 4-8 擴充詞庫前後之分析結果(J-AFINN 詞庫擴充後)

圖 4-8 是 J-AFINN 在詞庫擴充後的分析結果，整體準確率約有 79%，正負向方面的正確率約有 95%，正負向方面且不考慮中間分數的正確率約有 99%。

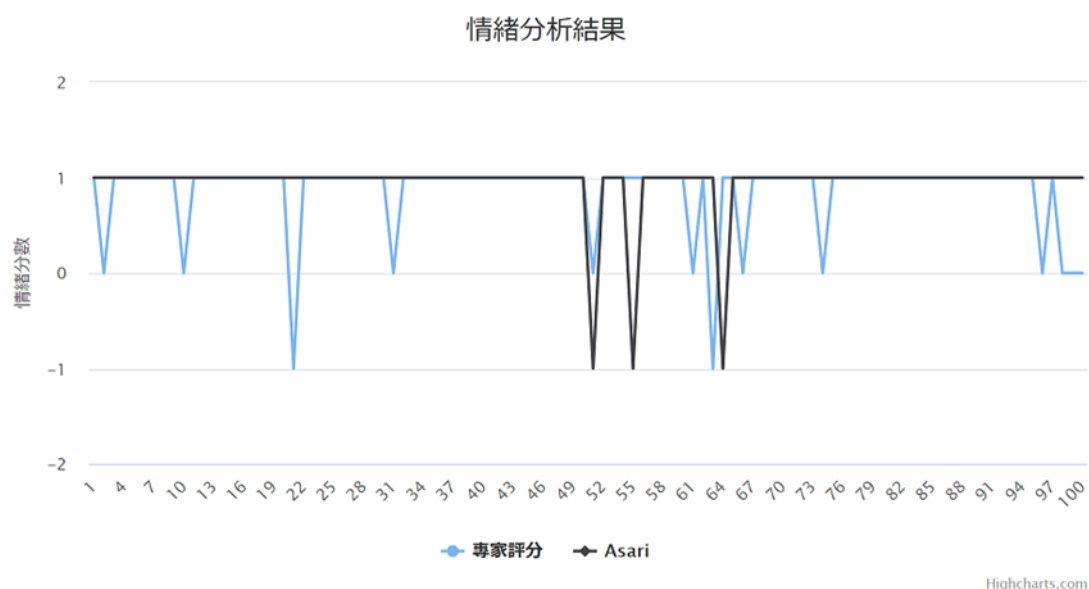


圖 4-9 擴充詞庫前後之分析結果(Asari, 無詞庫擴充)

圖 4-9 是 Asari 的分析結果，正負向方面的正確率約有 85%，正負向方面且不考慮中間分數的正確率約有 96%。

從以上分析結果可以看出 J-AFINN 在擴充後的準確率比擴充前高很多，證

實擴充詞庫對於 AFINN 這類使用情緒詞庫進行情緒分析的情緒分析方法確實可提高情緒分析的準確率。然而 J-AFINN 在擴充前的準確率與 Asari 相比就低了一點，Asari 這類使用類神經模型進行情緒分析的情緒分析方法在經過訓練之後就能維持一定的準確率，就如 Asari 的準確率都維持在約 80% 左右。

### 4.3 結合機器學習之分析結果

最後，本研究要探討 J-AFINN 結合機器學習演算法是否可提高分析準確率。預設不考慮 J-AFINN 的評價分數為 0 分的評價資料。為了與擴充詞庫的方法做比較，因此使用的是未擴充詞庫的 J-AFINN。評價資料方面使用日本酒全 200 筆評價資料和啤酒評價資料 100 筆。所使用的情緒分析方法和評價資料如表 4-3 所示，整體分析結果如圖 4-10 所示，以下針對個別分析結果作解析。

表 4-3 情緒分析方法與評價資料使用情況(結合機器學習之分析)

對照 1	情緒分析方法	J-AFINN (未擴充詞庫) + Asari
	評價資料	日本酒前 100 筆評價資料 (1 ~ 100)
對照 2	情緒分析方法	J-AFINN (未擴充詞庫) + Asari
	評價資料	啤酒評價資料 100 筆 (1 ~ 100)
對照 3	情緒分析方法	J-AFINN (未擴充詞庫) + Asari
	評價資料	日本酒後 100 筆評價資料 (101 ~ 200)

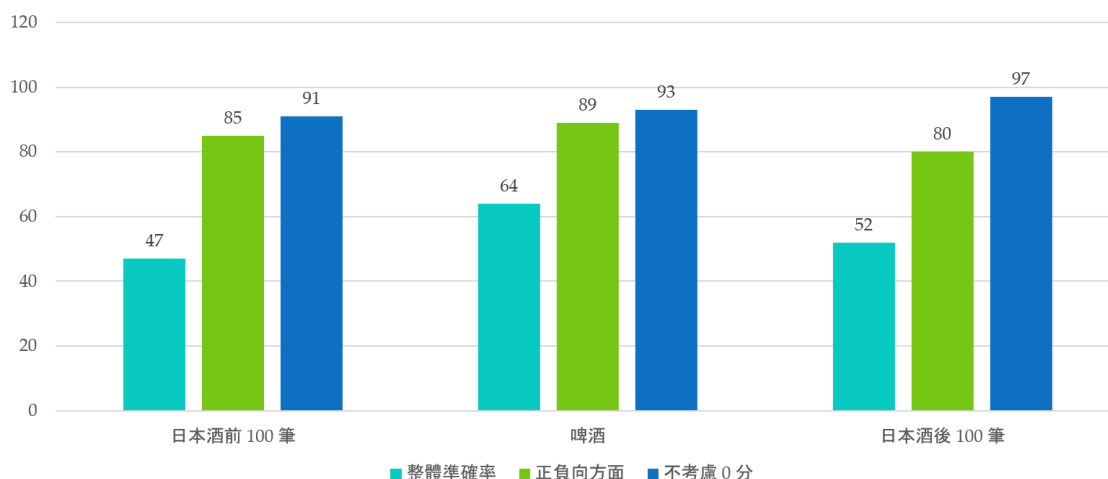


圖 4-10 結合機器學習之整體分析結果

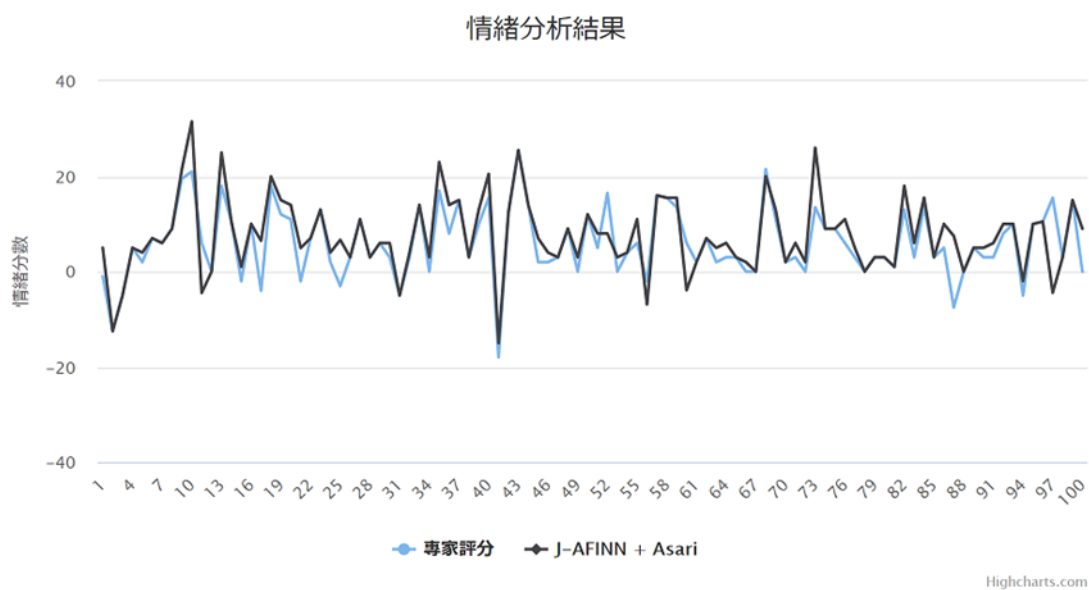


圖 4-11 結合機器學習之分析結果(日本酒前 100)

圖 4-11 是對日本酒前 100 筆評價資料的分析結果，整體準確率約有 47%，正負向方面的正確率約有 85%，正負向方面且不考慮中間分數的正確率約有 91%。

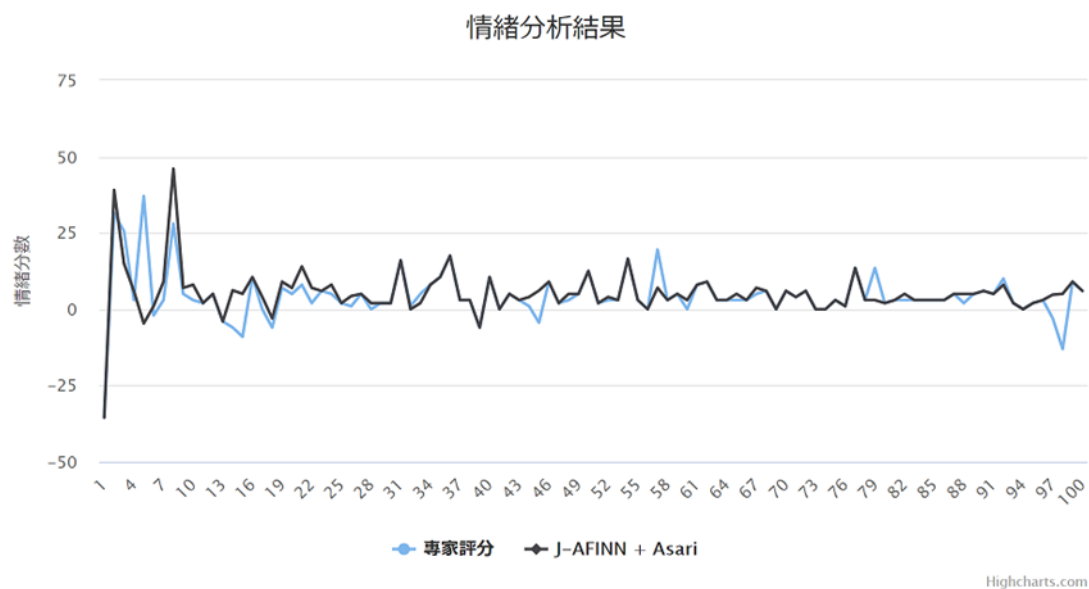


圖 4-12 結合機器學習之分析結果(啤酒)

圖 4-12 是對啤酒評價資料的分析結果，整體準確率約有 64%，正負向方面的正確率約有 89%，正負向方面且不考慮中間分數的正確率約有 93%。

與一般的 J-AFINN 相比沒有太大的差距，且準確率些微下降。在 4-1 的日本酒評價資料的分析結果中，J-AFINN 在正負向方面且不考慮中間分數的正確率約有 97%，而 Asari 則是 91%，可以得知兩者結合後的分析結果受到 Asari 的影響，因此正確率些微下降，在對啤酒評價資料的分析結果方面亦是如此。

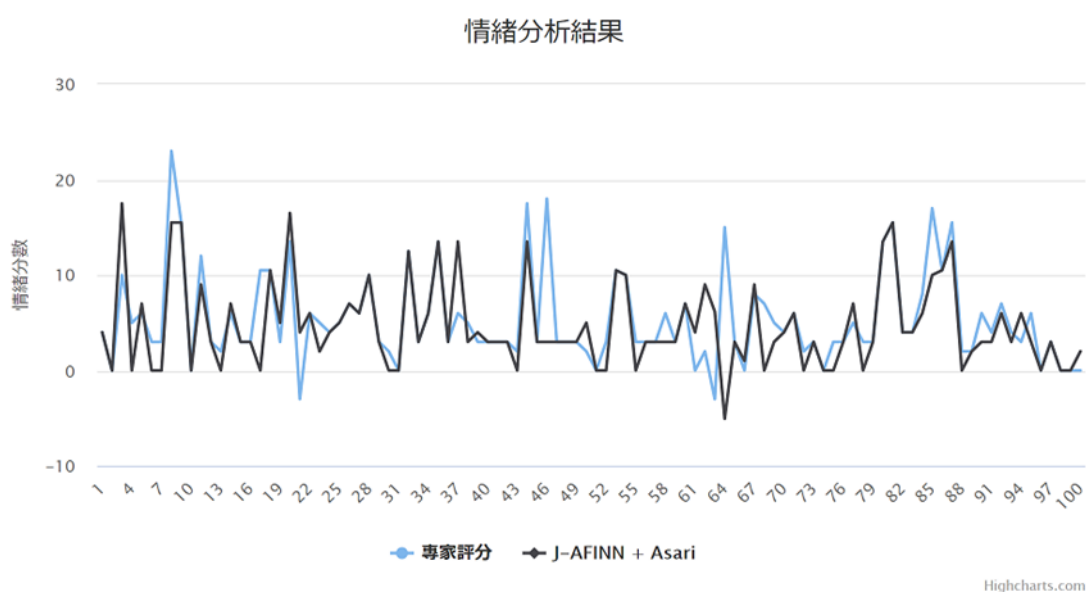


圖 4-13 結合機器學習之分析結果(日本酒後 100)

圖 4-13 是對日本酒後 100 筆評價資料的分析結果，整體準確率約有 52%，正負向方面的正確率約有 80%，正負向方面且不考慮中間分數的正確率約有 97%。

對日本酒後 100 筆評價資料的分析結果與前兩個分析結果也是一樣的情況，但有些分析結果因為詞庫未擴充，J-AFINN 無法正確評分導致分數為 0 分，因此這些分析結果沒有被考慮進去。在前兩個分析結果中不考慮 J-AFINN 的評價分數為 0 分的評價資料，是因為導致 J-AFINN 的評價分數為 0 分的原因並不會因為詞庫未擴充而造成。

為了與擴充詞庫的方法做比較，圖 4-14 是使用與圖 4-13 相同的評價資料，但有考慮 J-AFINN 的評價分數為 0 分的分析結果。

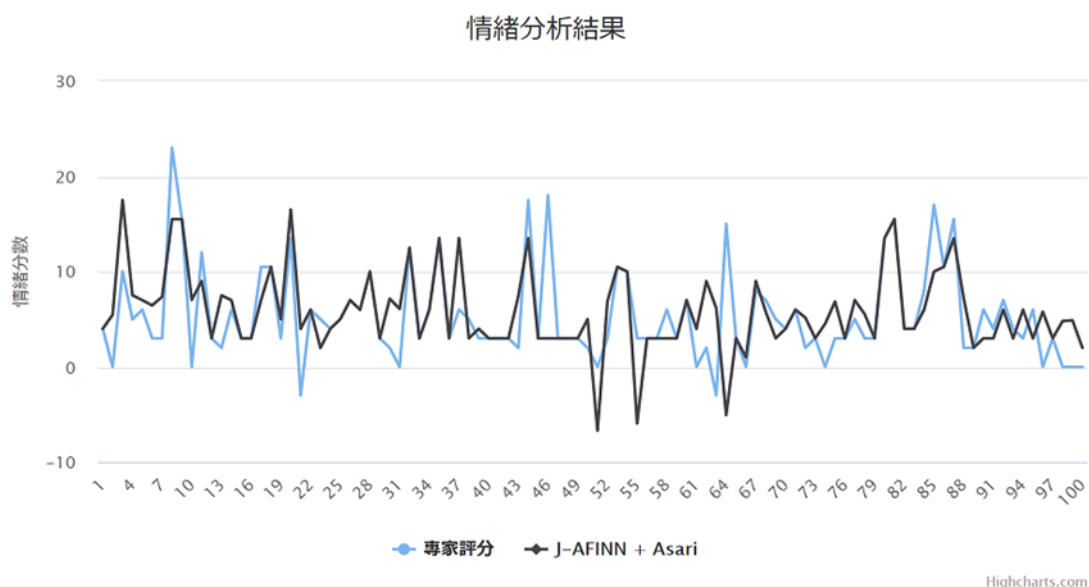


圖 4-14 結合機器學習之分析結果(日本酒後 100, 調整實驗條件)

雖然整體準確率降為約 44%，但正負向方面的正確率則提高至約 85%，正負向方面且不考慮中間分數的正確率只下降約 1% 且為 Asari 分析錯誤的結果因此忽略不計。在 4-2 的分析結果中擴充詞庫前後正負向方面的正確率分別是 82% 和 95%，與擴充詞庫後的分析正確率相比兩者結合後的分析正確率並不高，但與擴充詞庫前的分析正確率則提高約 3%，就整體而言雖然提升的效果並不高，但還是有效果。詳細分析結果如圖 4-15。

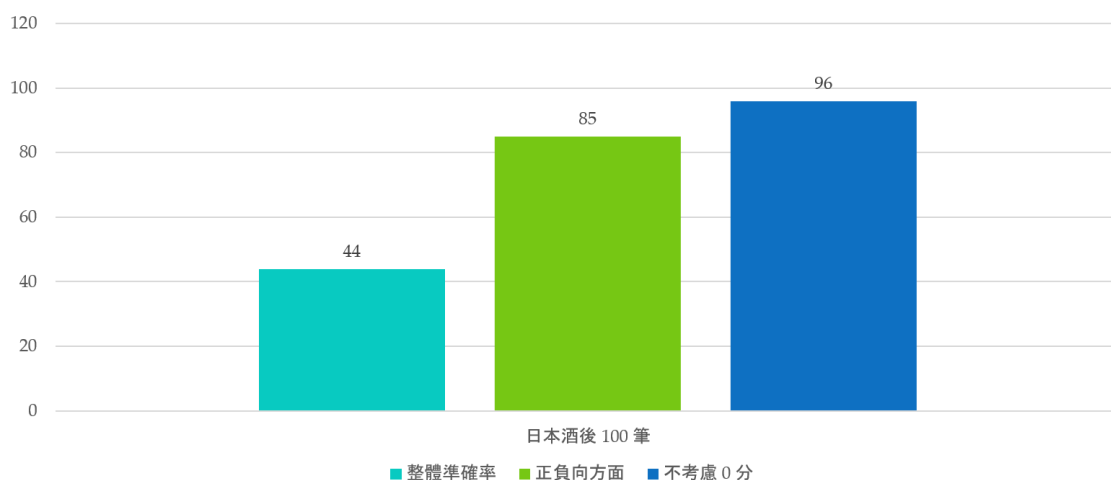


圖 4-15 結合機器學習之分析結果(日本酒後 100, 調整實驗條件, 長條圖)

# 第五章 結論與未來建議

## 5.1 結論

根據研究結果可以歸納出以下四點結論：

1. 傳統情緒分析方法配合情緒詞庫可針對特定評論範圍做優化，提高分析準確率。
2. 使用情緒詞庫的傳統情緒分析方法可透過擴充詞庫提高分析準確率。
3. 機器學習情緒分析方法透過類神經模型進行情緒分析，不會因詞庫而影響分析準確率，且有穩定的分析準確率。
4. 利用機器學習情緒分析方法的穩定的分析準確率可改善使用情緒詞庫的傳統情緒分析方法在擴充詞庫前正負向方面分析正確率不佳的問題。

傳統情緒分析方法與機器學習情緒分析方法各有優缺點，傳統情緒分析方法配合情緒詞庫可在不同評論範圍有良好的分析準確率且可透過擴充詞庫進一步改善，但擴充詞庫需依靠人力。機器學習情緒分析方法使用的類神經模型經過訓練後有穩定的分析準確率，但無法針對不同評論範圍做優化。

傳統情緒分析方法與機器學習情緒分析方法的結合，除了保有情緒詞庫可針對不同評論範圍做優化以及可擴充性的優點外，機器學習情緒分析方法的穩定的分析準確率可在詞庫擴充前的情況下保有一定的分析準確率。此外透過兩者的結合，找出兩者分析結果相異的評價資料作為後續改進演算法以及擴充詞庫的依據，可減輕人力的負擔。

## 5.2 未來研究建議

本研究僅針對酒類商品的評價資料作分析，未來還可以對其他種類商品（如食品類）的評價資料作分析，測試兩種情緒分析方法在其他種類商品的評價資料上的分析結果。本研究提出傳統情緒分析方法配合情緒詞庫可對多個評論範圍做優化，

以及機器學習情緒分析方法可維持一定的分析準確率兩個說法，未來也可以針對多種不同種類商品的評價資料作分析，測試兩種情緒分析方法在多種不同種類商品的評價資料上是否保持各自的優勢。



# 參考文獻

## 中文參考文獻

- 李淑惠 (2014)。運用文字探勘技術於口碑分析之研究 (碩士論文)。
- 林彩雯 (2015)。以 Google App 評論為字詞權重調整之情緒分析系統 (碩士論文)。
- 施威銘研究室 (主編) (2019)。tf.keras 技術者們 - 必讀！深度學習攻略手冊。  
臺北市：旗標。
- 陳安怡 (2017)。運用文字探勘及情緒分析技術發展店家品項評價模組 (碩士論文)。
- 游和正、黃挺豪、陳信希 (2012)。領域相關詞彙極性分析及文件情緒分類之研究。  
中文計算語言學期刊，17(4)，33-47。
- 葉欣睿 (譯) (2019)。Deep learning 深度學習必讀 - Keras 大神帶你用 Python 實作 (原作者：F. Chollet)。臺北市：旗標。(原著出版年：2017)
- 謝梁、魯穎、勞虹嵐 (2018)。Python 深度學習實作：Keras 快速上手。新北市：博碩文化。
- 藍子軒 (譯) (2019)。TensorFlow 自然語言處理：善用 Python 深度學習函式庫，  
教機器學會自然語言 (原作者：G. Thushan)。臺北市：基峰資訊。

## 英文參考文獻

- Hotho, A., Nürnberger, A., Paaß, G. (2005). A Brief Survey of Text Mining.
- Nielsen, F. Å. (2011). A new ANEW: Evaluation of a word list for sentiment analysis in microblogs.
- Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., Kappas, A. (2010). Sentiment Strength Detection in Short Informal Text. Journal of the American Society for Information Science and Technology, 61(12), 2544-2558. doi: 10.1002/asi.21416

Yousef, A. H., Medhat, W., Mohamed, H. K. (2014). Sentiment Analysis Algorithms and Applications: A Survey.

## 日文参考文献

勝間田昇、山岸直秀、隈裕子、鈴木誠（2019）。ゴルフ場のレビューデータを用いた感情極性辞書の作成。経営情報学会 2019 年春季全国研究発表大会，133-136。

木村真有、桂井麻里衣（2016）。ソーシャルメディアを用いた絵文字の感情極性分類。2016年度情報処理学会関西支部 支部大会。

竹内広宜、大野正樹（2018）。顧客コメントからのプロフィール情報の抽出および製品・サービス情報の構造化を利用した商品推薦システム。情報処理学会論文誌データベース，10(2)，42-50。

中山貴樹（2012）。構文構造を用いたテキスト感情極性分析の精度向上（碩士論文）。

森田晋也、白井靖人（2018）。分野別感情極性辞書の作成及び評価。情報処理学会第 80 回全国大会，317-318。

山内崇資、林佑樹、中野有紀子（2013）。日本語文解析によるTwitterの感情分析とシーンインデキシングへの応用。情報処理学会第 75 回全国大会，315-316。