

你是什麼意思？😂😳😡

— 表情符號推薦器

第七組：

林貝爾、陳姸蓁、易祐辰、黃亮臻、黃淑郁、彭琳



報告架構

- 動機與目的
- 文獻回顧
- 資料收集與預處理
- 研究方法
- 研究分析與結果
- 討論與後續研究建議



研究動機與目的

動機

人們使用**通訊軟體**進行**文字訊息**傳遞時，經常會在語句中加上對應的表情符號。

目的

探討一段句子被人所賦予的**情緒**，並找出最貼近這些文字情緒的**表情符號**。



文獻回顧－說明

林育龍先生

《對使用者評論之情感分析研究－以 Google Play 為例》

- 針對app市集評論進行中文文本之文字探勘
- 利用中文情感分析判斷一個評論的正負向情緒

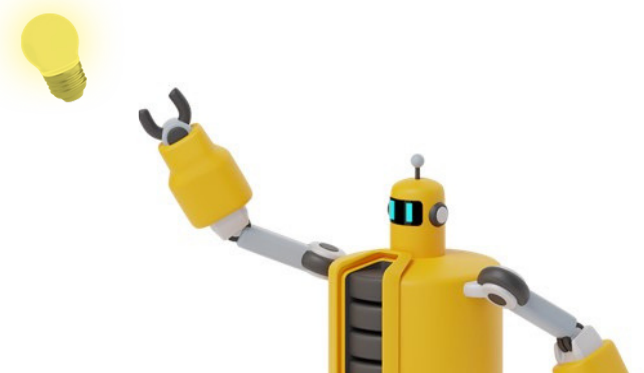
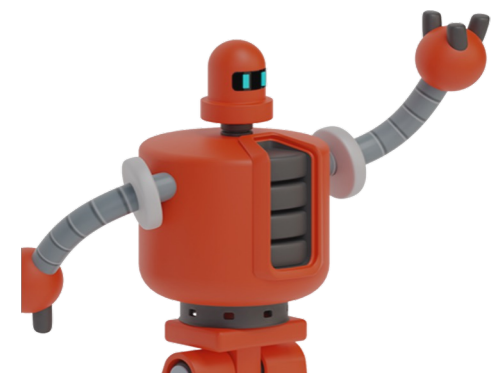
文獻回顧－特色差異

林育龍先生論文

- 為2014年之研究結果
- 針對app評論相關之文本
- 判別正負向情緒

我們的研究

- 範圍為2018-2022年間
- 類型涵蓋不同面相
- 將情緒細分成更多類型



資料收集與預處理 – 建立表情符號分類模型之資料

Facebook 公開專業及社團



- 共爬取 407 個頁面
- 約 46 萬筆帶有表情符號資料

45.34%

Line 聊天紀錄



- 共收集 73 個聊天室
- 約 2 萬筆帶有表情符號資料

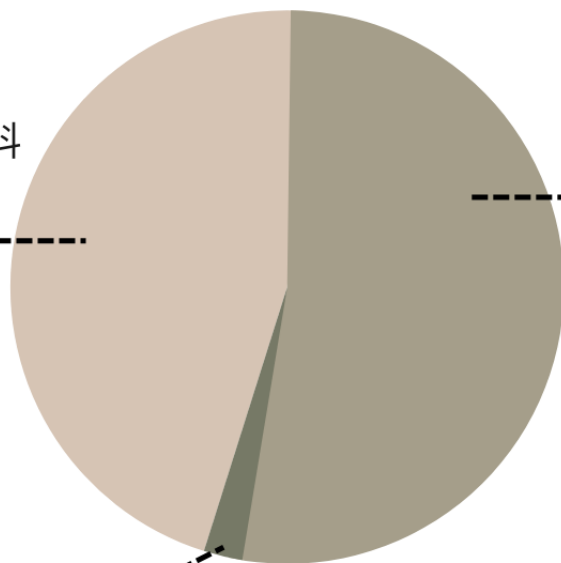
2.26%

Instagram 公開帳號

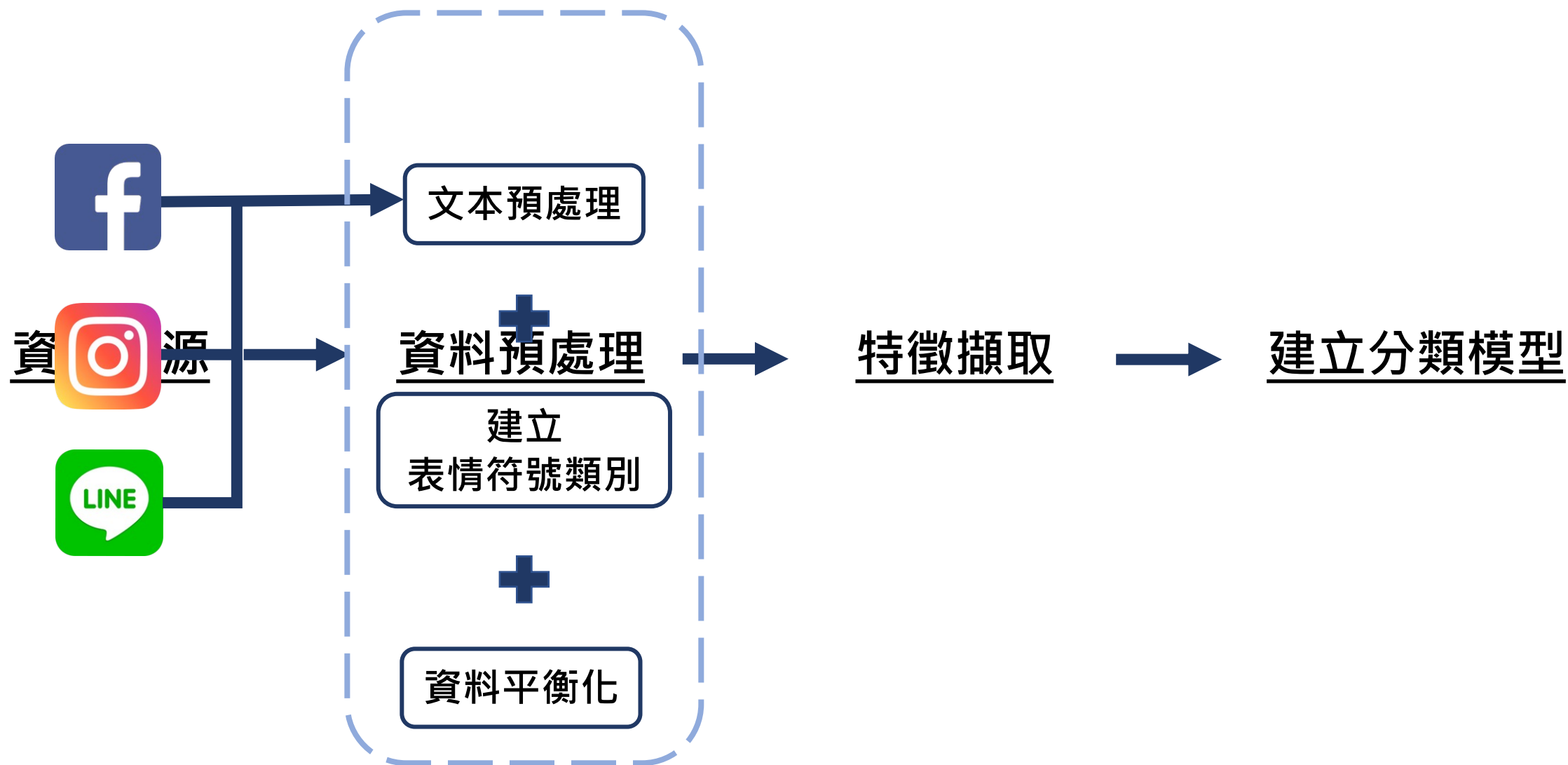


- 共爬取 244 個頁面
- 約 53 萬筆帶有表情符號資料

52.39%



資料收集與預處理 – 建立表情符號分類模型之資料



(一) 文本預處理

1. 爬蟲



2. 斷句



3. 斷詞

自己是長這樣



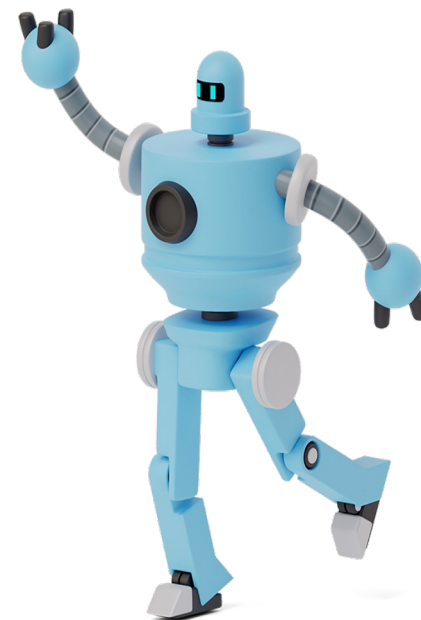
我很欣賞
我很欣賞



自己, 是, 長, 這樣


























我, 很, 欣賞
我, 很, 欣賞



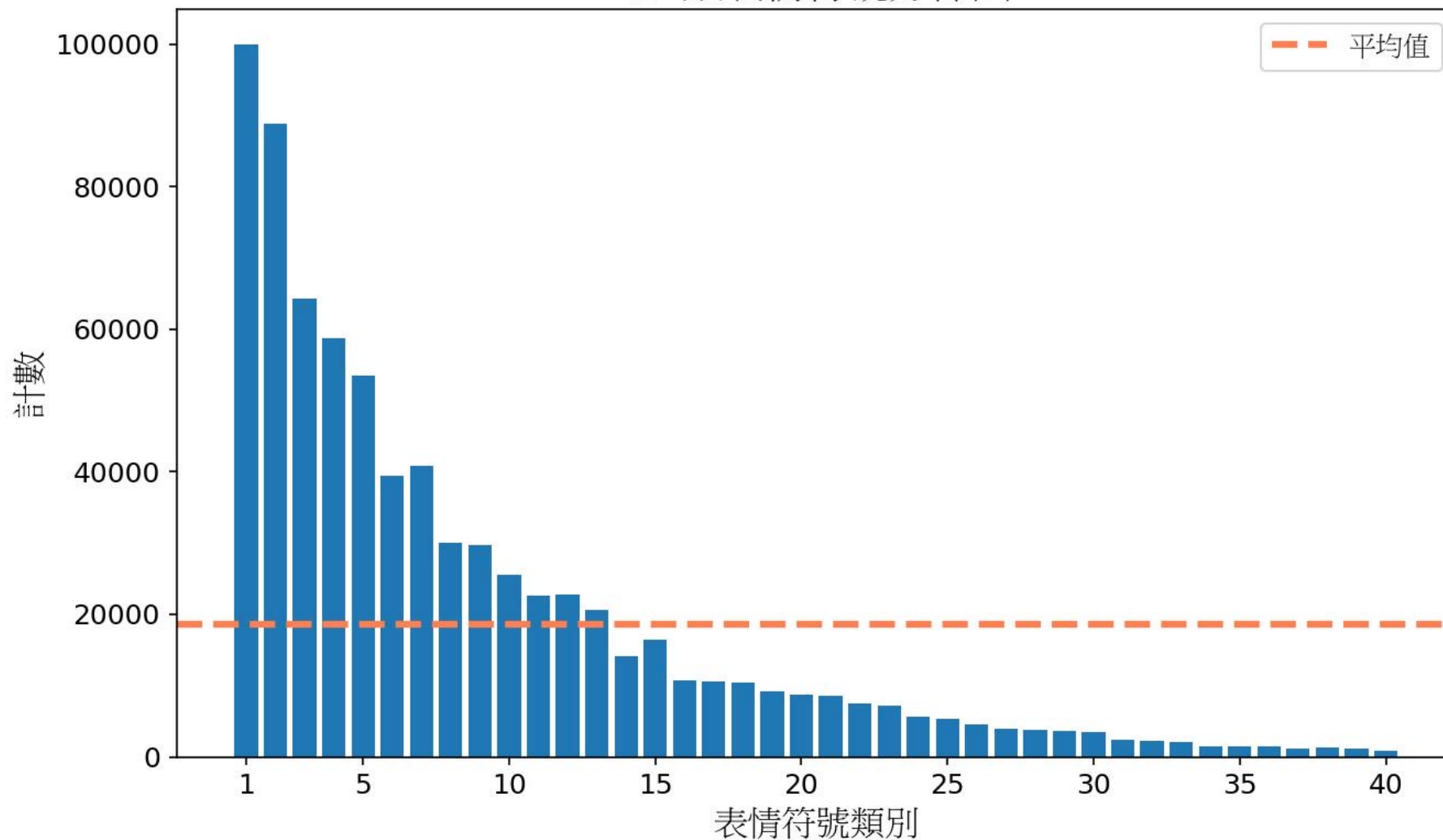
(二) 建立表情符號類別

- 取得資料樣本包含1020個表情符號
- 篩選出帶有情緒意涵的214個表情符號
- 合併有相似情緒與使用時機的表情符號，分為40類

1  	2  	3      	4 	5        	6    
--	--	--	--	--	---

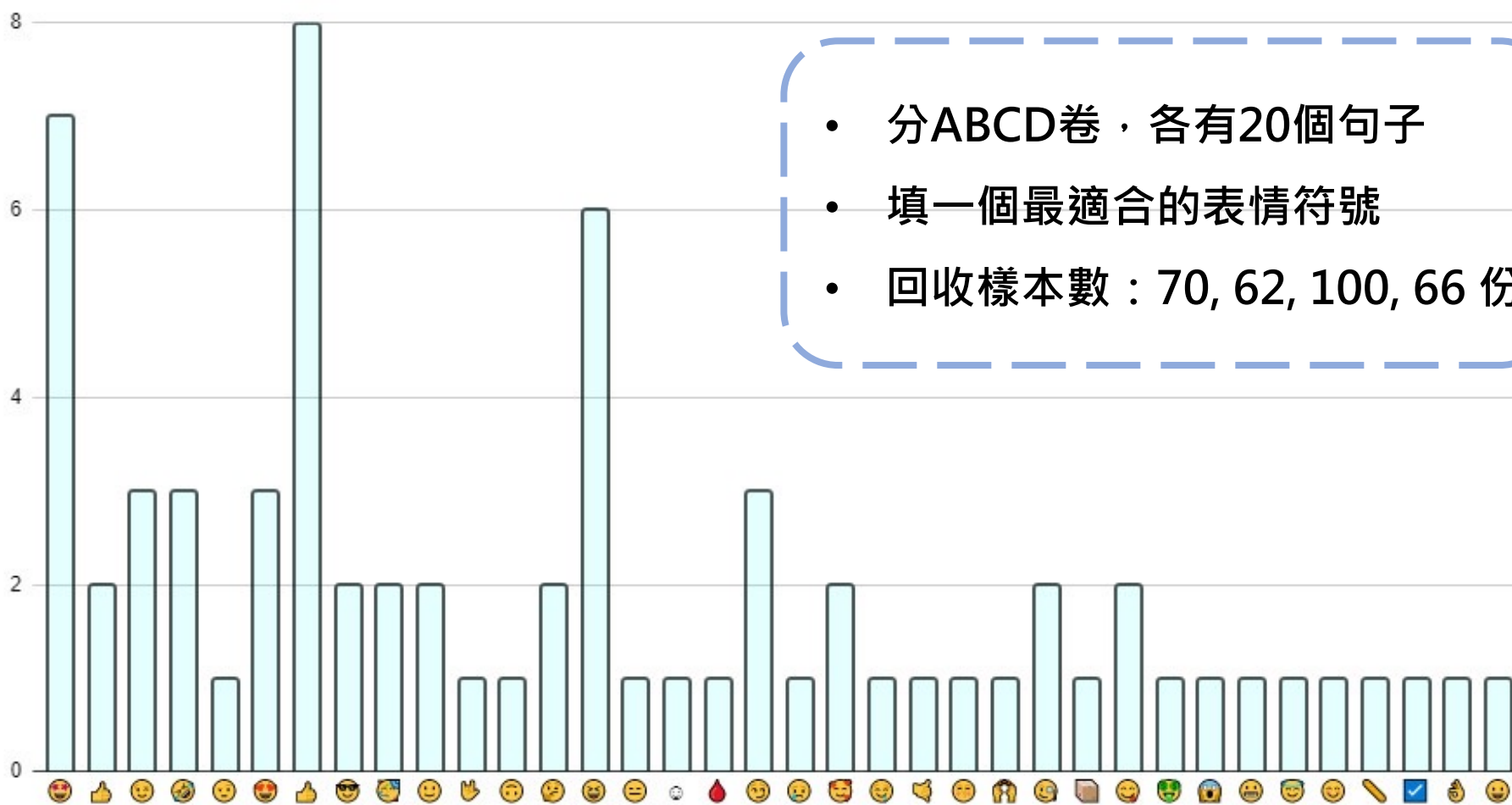
(三) 資料平衡化

40 類表情符號分佈圖



資料收集與介紹 – 與模型推薦表現比較之問卷調查

題目範例：真的精彩



研究方法

資料來源

資料預處理

特徵擷取



文本預處理



建立
表情符號類別



資料平衡化

有emoji資料

建立
One Hot
Encoding

LASSO
選詞

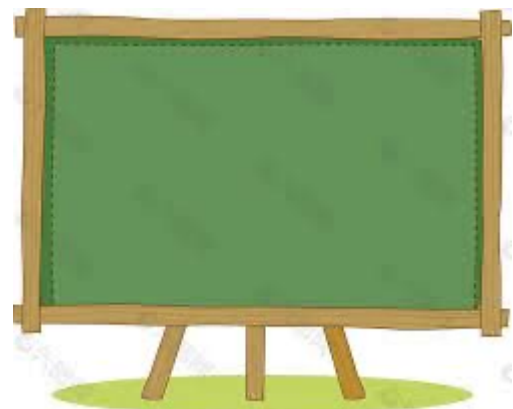
刪去emoji資料

詞嵌入
Word2Vec
計算詞向量

TFIDF
計算句向量

One-Hot-Encoding

舉例：「專題一起努力。」



專題
一起
努力

One-Hot-Encoding



100

010

001

LASSO選詞

原因：

- 原始資料有9萬個不同詞彙
- 資料量龐大且矩陣太過稀疏

優點：

- 不重要的詞彙所對應的係數會收縮到0
- 篩選出較重要的詞彙



研究方法

資料來源

資料預處理

特徵擷取



文本預處理



建立
表情符號類別



資料平衡化

有emoji資料

建立
One Hot
Encoding

LASSO
選詞

刪去emoji資料

詞嵌入
Word2Vec
計算詞向量

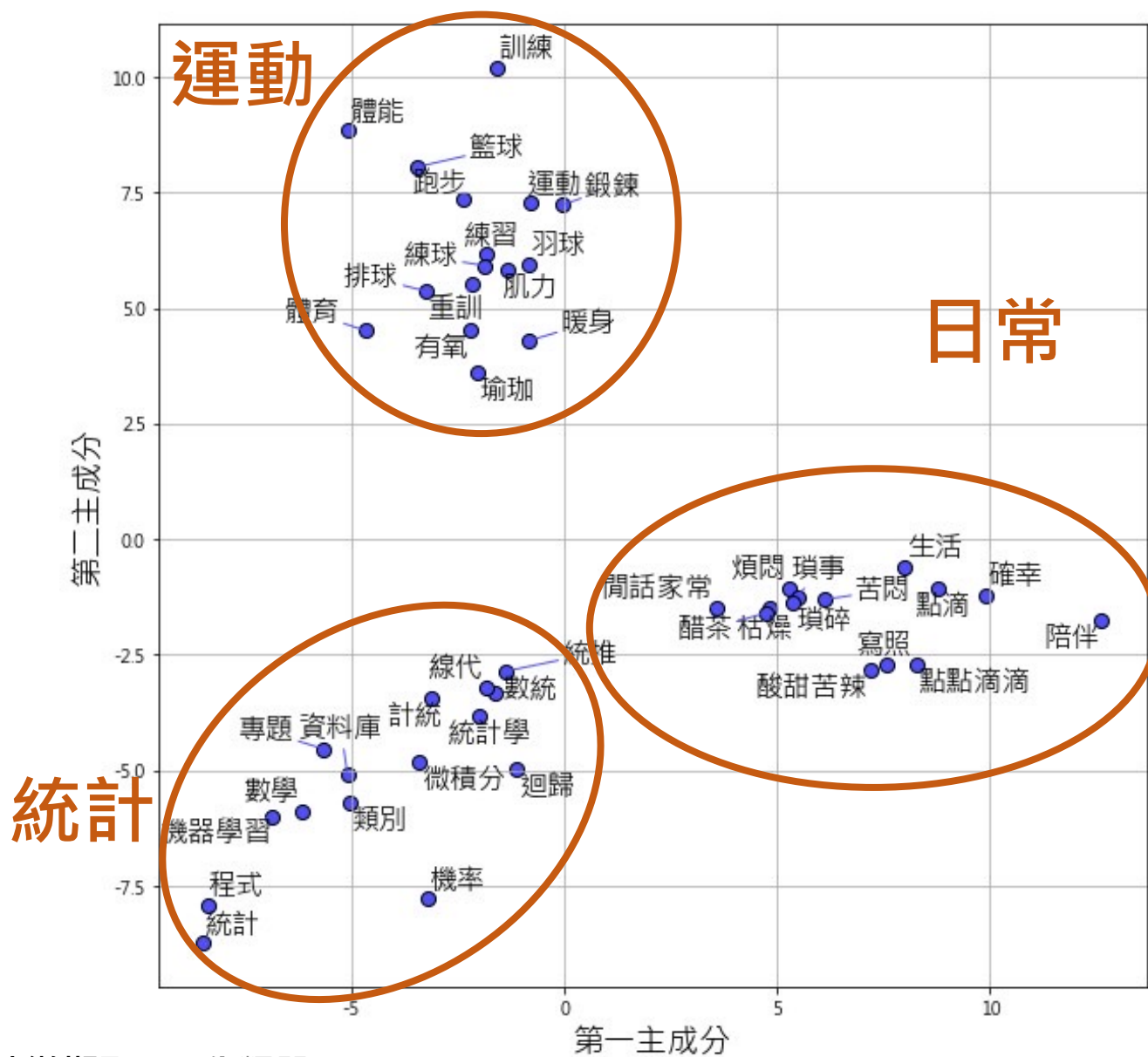
TFIDF
計算句向量

Word2Vec

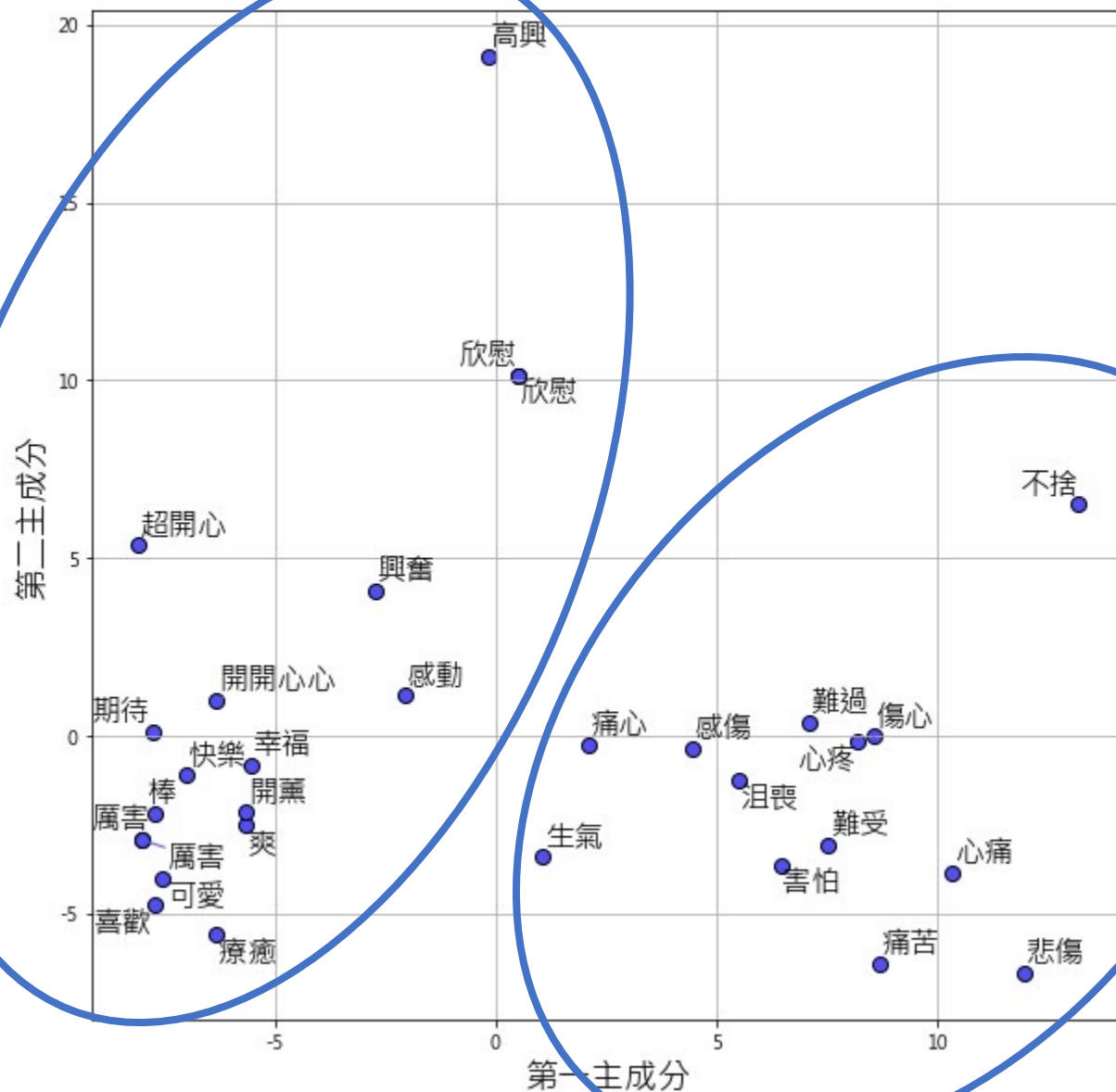
將單詞轉為詞向量，找出詞與詞之間的相關性



使用 PCA 將詞向量視覺化



詞向量與情緒有關的視覺化圖



句向量

使用 **TF-IDF** 演算法對詞進行加權平均。

- TF：詞在一份文件出現的次數，即為詞頻
- IDF：包含詞的文件比例

TF×IDF值越大，給予該詞彙較大**權重**



句向量



舉例：「我明天想要去慶生欸。」

$$\text{句向量} = \frac{0.1 \times \text{我} + 0.15 \times \text{明天} + 0.2 \times \text{想要} + 0.15 \times \text{去} + 0.4 \times \text{慶生} + 0.1 \times \text{欸}}{6}$$

研究方法

資料來源 資料預處理



文本預處理



建立
表情符號類別



資料平衡化

資料特徵擷取

One Hot
Encoding
↓
LASSO選詞

Word2Vec
詞向量
↓
TFIDF
句向量

建立分類模型

Multinomial Naïve Bayes

KNN

SVM

Guassian Naïve Bayes

分類器種類



- KNN：多數決計算相鄰個數
- SVM：找出決策邊界讓兩類之間 margin 最大化
- Naïve Bayes：利用貝式定理找出發生最大機率的類別

推薦表情符號，設定3個中1個/ 5個中1個即正確

分析結果

Multinomial LASSO 搭配 Multinomial Naïve Bayes

訓練集 編號	特徵選取	平衡化	樣本數	變數數量	測試集 正確率 (3中1)	測試集 正確率 (5中1)
1.1	無	無	150,000	9,808詞彙	56.1%	66.6%
1.2	LASSO	無	665,000	2,587詞彙	56.9%	68.9%
1.3	無	有	58,652	9808詞彙	23.9%	38.6%
1.4	LASSO	有	58,652	2,587詞彙	38.8%	48.7%



分析結果

Multinomial LASSO 搭配 Multinomial Naïve Bayes

訓練集 編號	特徵選取	平衡化	樣本數	變數數量	測試集 正確率 (3中1)	測試集 正確率 (5中1)
1.1	無	無	150,000	9,808詞彙	56.1%	66.6%
1.2	LASSO	無	665,000	2,587詞彙	56.9%	68.9%
1.3	無	有	58,652	9808詞彙	23.9%	38.6%
1.4	LASSO	有	58,652	2,587詞彙	38.8%	48.7%

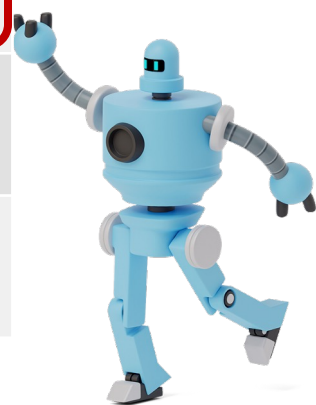


分析結果

句向量資料搭配 KNN、SVM、Gaussian Naïve Bayes

利用 PCA 將句向量300維降至前 20 個主成分

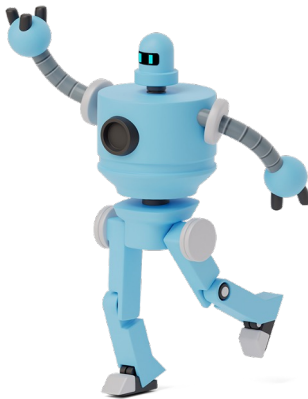
Accuracy	不平衡		平衡	
	測試集 (3中1)	測試集 (5中1)	測試集 (3中1)	測試集 (5中1)
SVM	53.3%	65.6%	36.4%	46.8%
KNN	55.0%	66.4%	34.8%	44.8%
Gaussian	45.8%	57.0%	28.6%	39.3%



分析結果

模型表現與問卷結果之比較

Accuracy	問卷(3中1)		問卷(5中1)	
	不平衡	平衡	不平衡	平衡
SVM	35.0%	33.8%	40.0%	41.3%
KNN	20.0%	23.8%	25.0%	37.5%
Gaussian	20.0%	10.0%	30.0%	20.0%
問卷	20%		25%	



討論與後續建議

一、資料不平衡再處理

方法1：Tomek Link (欠採樣 undersampling)

方法2：SMOTE (過採樣 oversampling)

方法3：SMOTE + ENN



討論與後續建議

一、資料不平衡再處理

結果：模型的表現皆比原始不平衡資料差

40類 正確率	原KNN	Tomek links	SMOTE	SMOTE + ENN
測試集 3中1	55.0%	54.9%	44.1%	14.6%
測試集 5中1	66.4%	66.0%	54.1%	14.6%



討論與後續建議

二、表情符號類別數修正

方法：類別數調降成20組

Accuracy	40類 3中1	20類 3中1	40類 5中1	20類 5中1
問卷	20.0%	35.0%	25.0%	42.5%
KNN	55.0%	59.9%	66.4%	72.5%
SVM	53.3%	58.4%	65.6%	71.5%
GauBayes	45.8%	50.7%	57.0%	64.0%
MultBayes	56.1%	60.3%	66.9%	72.5%

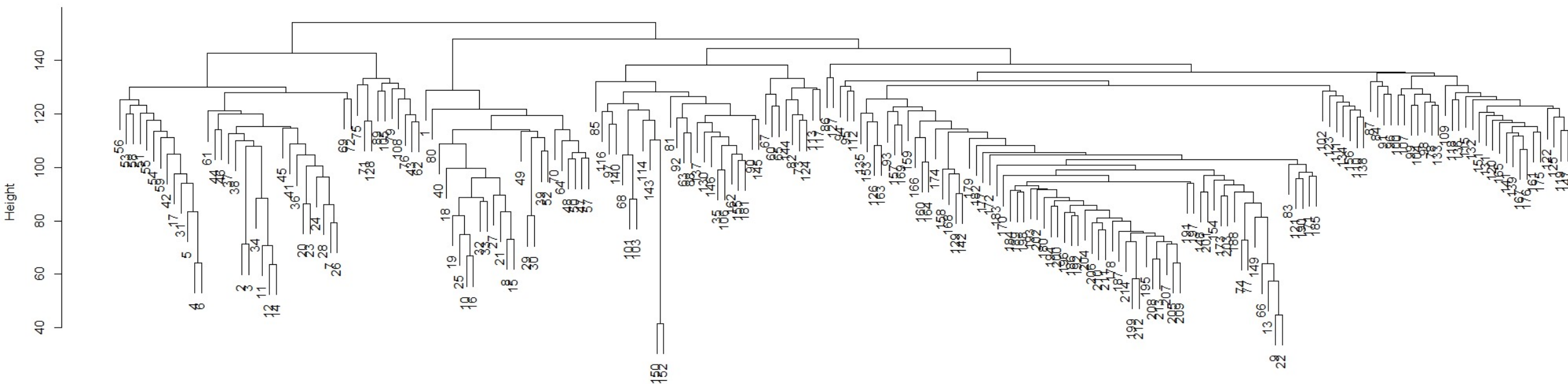
討論與後續建議

三、表情符號類別修正

方法：由**資料特徵**（而非人為判別）建立表情符號類別

結果：使用 KNN 預測五大類的準確率為99.9%

214個表情符號之分群樹狀圖

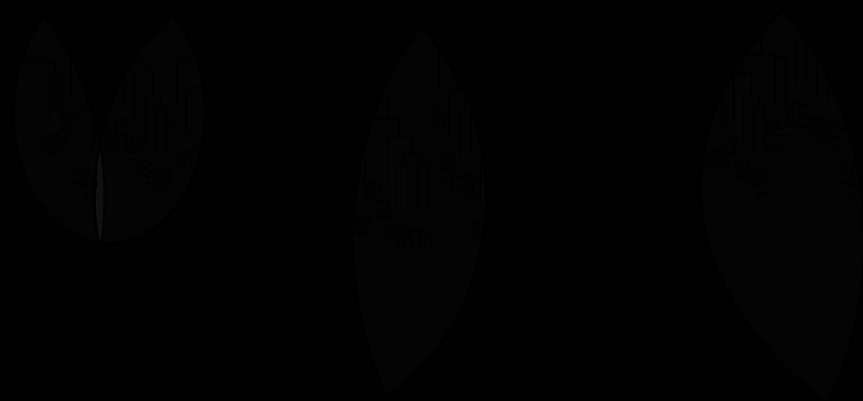


討論與後續建議

三、表情符號類別修正

方法：由資料特徵（而非人為判別）建立表情符號類別

結果：使用 KNN 預測五大類的準確率為99.9%



討論與後續建議

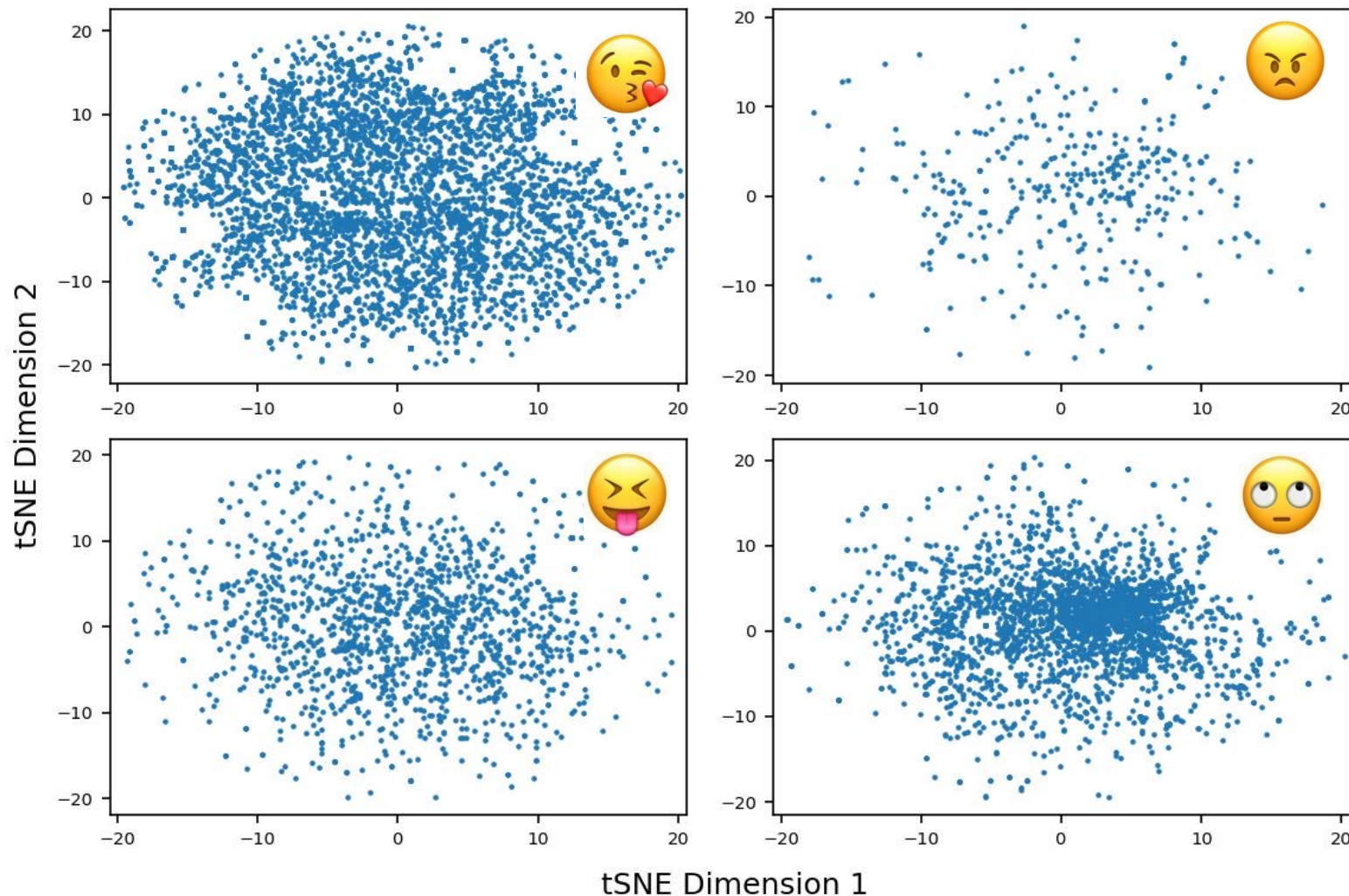
四、檢視句向量

300維句向量



t-SNE 二維散佈圖

句向量的資訊
可能不足以辨別
表情符號類別

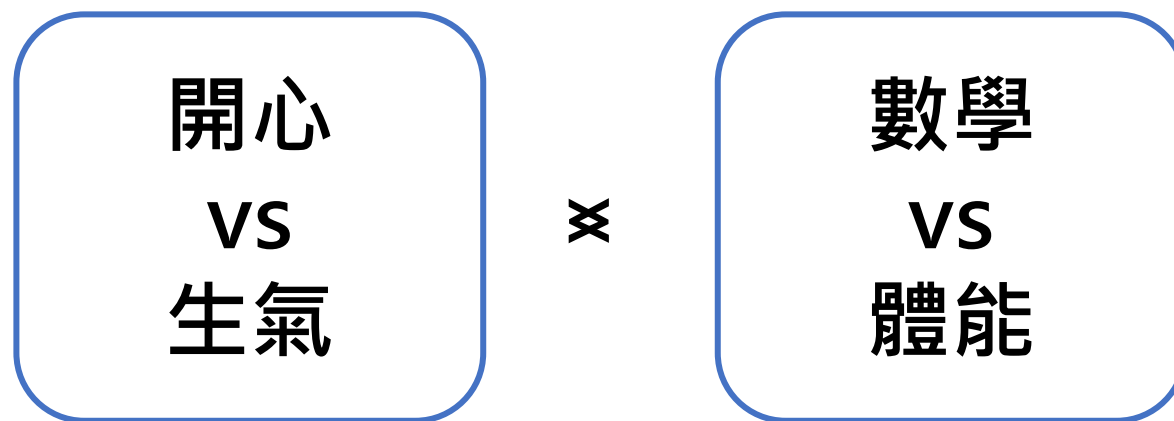


討論與後續建議

四、檢視句向量

原因：詞向量轉句向量的加權不理想

情緒影響：



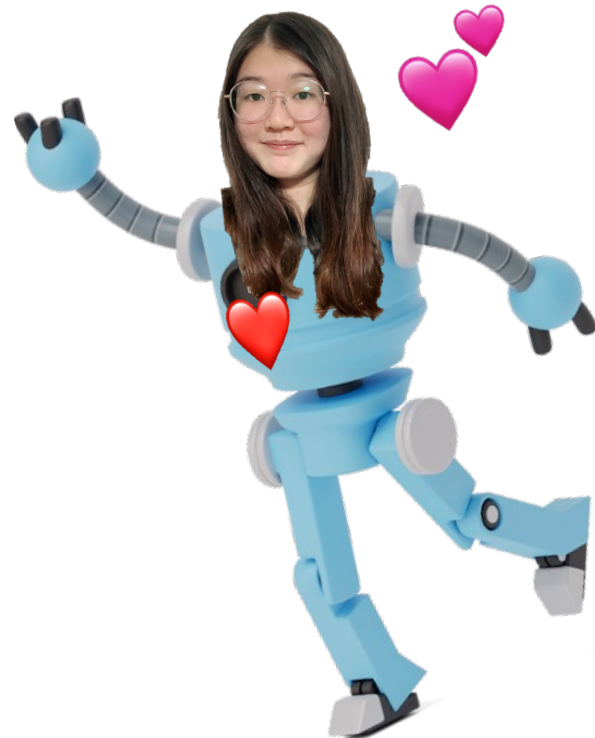
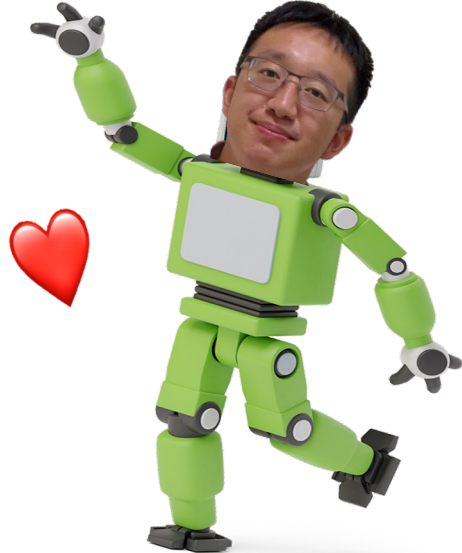
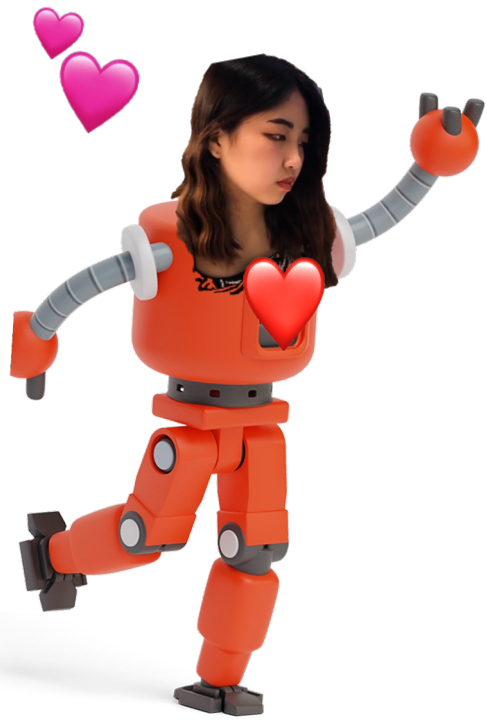
改善方法：

1. 使用LASSO對詞向量給予權重
2. 透過分群找出與情緒相關的詞，並給較高權重

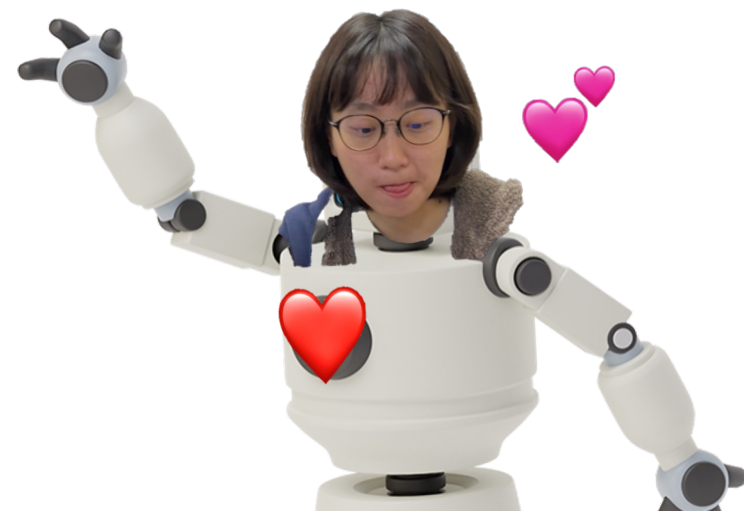
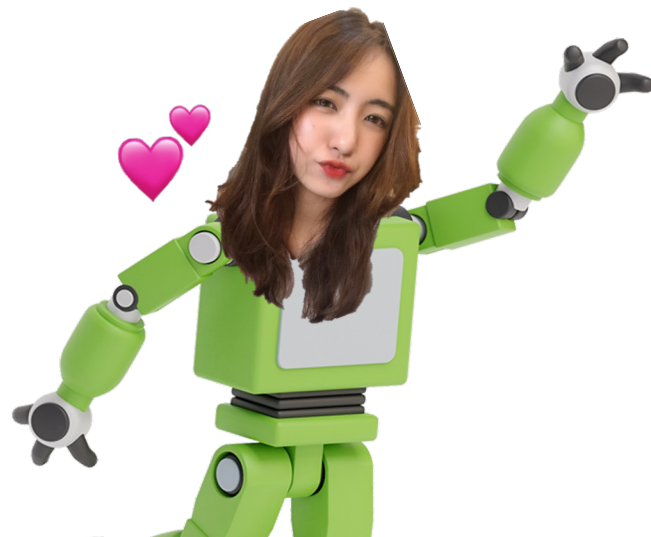
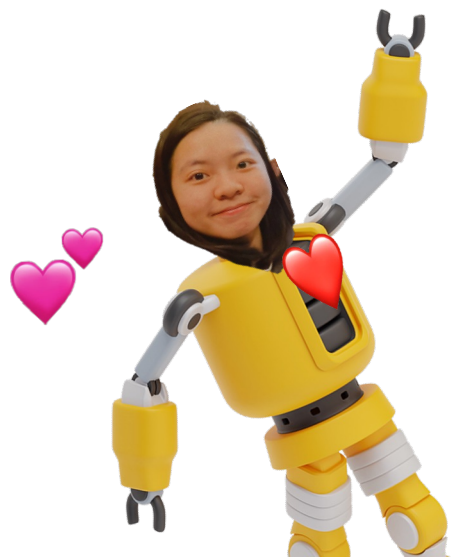
Choose One !!!

- 化妝師為什麼要這樣
- 數不清啦
- 身為芋頭控一定要去吃看看

1	2	3	4	5	6	7		
😍🐱	😭🐱	👍👏🏆👍🏆👍	😭🎉🍰🎉🍷🍷🍷	🍰🎉🍰🎉🍷🍷🍷	😭😭😭🐱🐱🐱	❤️❤️❤️❤️❤️❤️		
8	9	10	11	12	13	14	15	16
😍👩👩👩👩👩👩	😭😭😭🐱🐱🐱	💪📺	😭👁️👁️👁️👁️	🔥⚡	🐱👊👊👊	😭😭😭	😭👉👉👉	😭👉👉👉
👩👩👩👩👩👩	🐱🐱🐱🐱🐱🐱		👩👩👩👩👩👩		🐱🐱🐱🐱🐱🐱			
17	18	19	20	21	22	23	24	
😭😭	🙏	😭😭😭🐱🐱🐱	😭😭😭?	😭😭	😭😭😭👩	😭😭😭👩	😭👩	
		😭😭😭🐱🐱🐱	😭😭?		😭😭😭👩	😭😭😭👩		
		👩👩👩👩👩👩						
25	26	27	28	29	30	31	32	33
😭😭	👩👩👩👩👩👩	😭😭😭🐱🐱🐱	😭😭😭🐱🐱🐱	😭😭😭👩👩	👩👩👩👩👩👩	😭😭😭👩👩	👩👩👩👩👩👩	😭😭😭👩👩
	😭😭😭🐱🐱🐱	😭😭😭🐱🐱🐱	😭😭😭🐱🐱🐱	😭😭😭👩👩	👩👩👩👩👩👩	😭😭😭👩👩	👩👩👩👩👩👩	😭😭😭👩👩
	👩👩👩👩👩👩	😭😭😭🐱🐱🐱	😭😭😭🐱🐱🐱	😭😭😭👩👩	👩👩👩👩👩👩	😭😭😭👩👩	👩👩👩👩👩👩	😭😭😭👩👩
34	35	36	37	38	39	40		
😭😭💰💰👩👩	👩👩👩👩👩👩	😭😭😭🐱🐱🐱	❌❌❌❌❌❌	🐱🐱🐱🐱🐱🐱	😭😭😭👩👩	😭😭😭👩👩		
👩👩👩👩👩👩	👩👩👩👩👩👩	😭😭😭🐱🐱🐱	❌❌❌❌❌❌	🐱🐱🐱🐱🐱🐱	😭😭😭👩👩	😭😭😭👩👩		
		😭😭😭🐱🐱🐱	❌❌❌❌❌❌	🐱🐱🐱🐱🐱🐱	😭😭😭👩👩	😭😭😭👩👩		

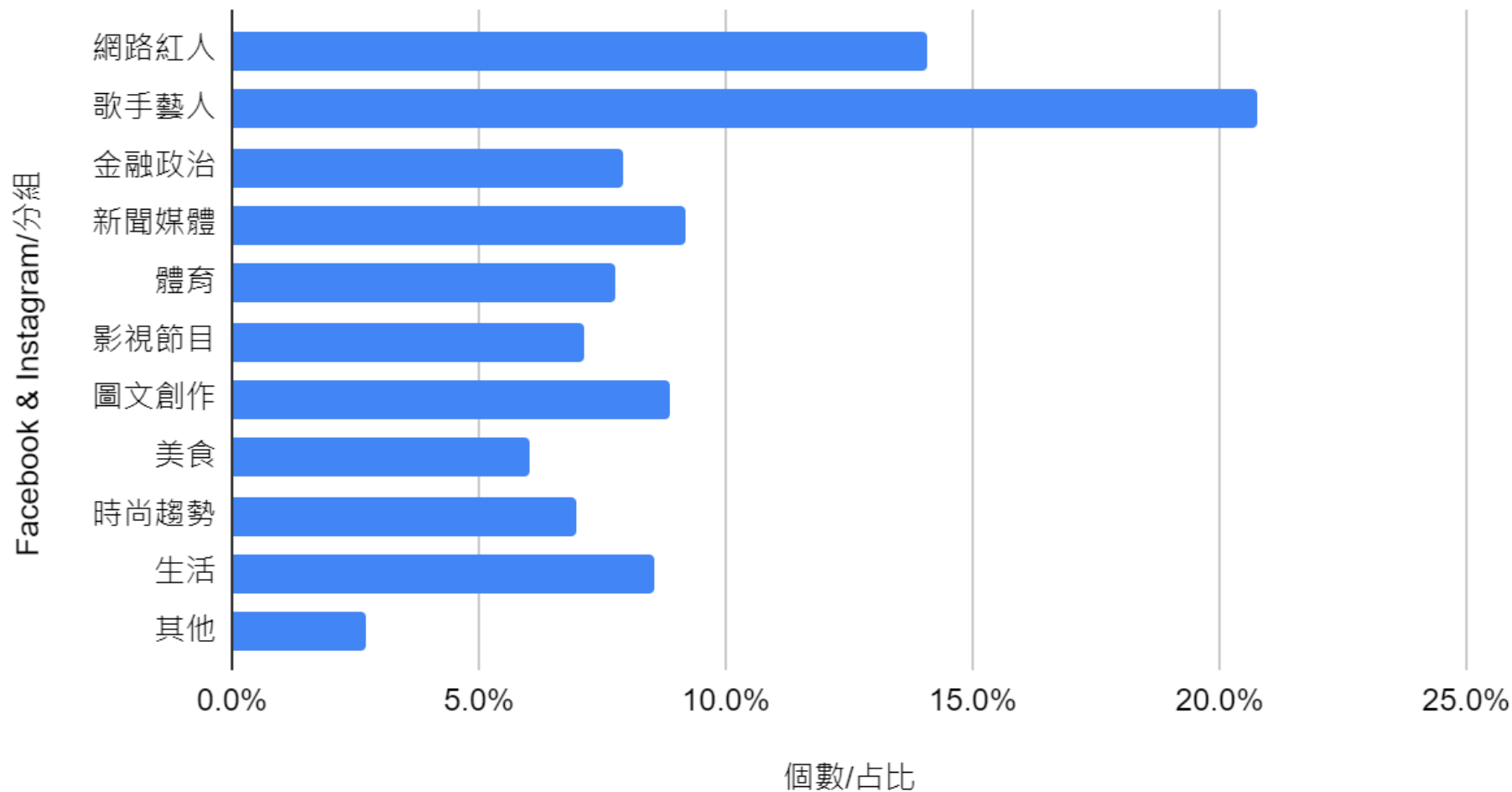


謝謝大家









































































































































































































































附錄1：目前已爬取之網站類型占比

縱軸：個數/占比，橫軸：Facebook & Instagram/分組



附錄2：表情符號

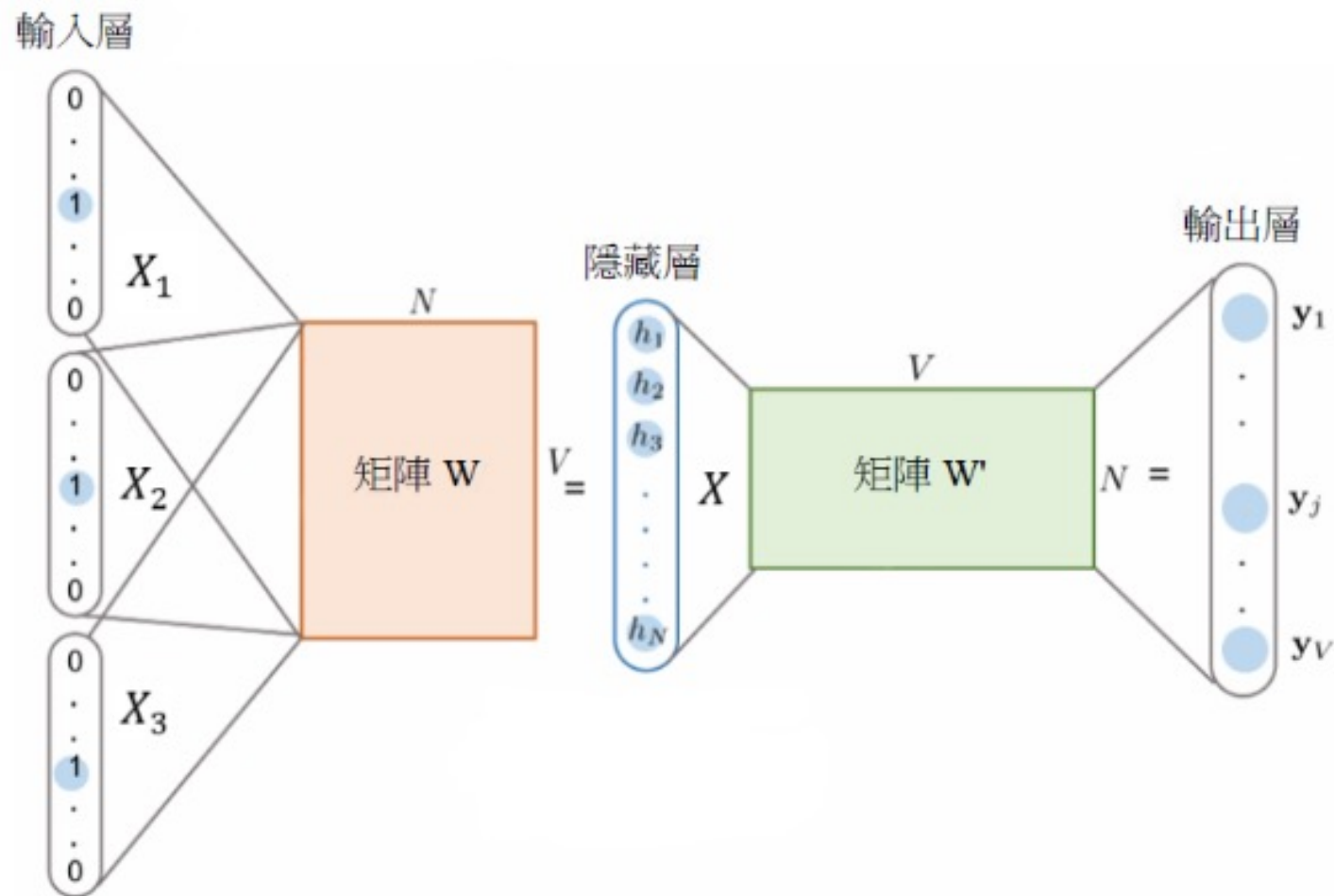
1  	2  	3       	4 	5           	6         	7                 		
8            	9    	10  	11      	12  	13         	14  	15 	16         
17  	18 	19            	20      	21  	22         	23         	24  	
25  	26         	27      	28         	29      	30         	31      	32         	33   
34      	35      	36    	37    	38    	39    	40    		

Word2Vec

常用參數名稱	值/方法
訓練模型方法(sg)	CBOW
加快訓練速度的方法(hs)	negative sampling
矩陣維度大小(vector_size)	300
模型訓練次數(epochs)	30
過濾少於此出現次數的詞(min_count)	10
詞產生上下文關係的個數(window size)	9

常用參數名稱	值/方法
訓練模型方法(sg)	0=CBOW

CBOW : 給定背景詞預測目標詞

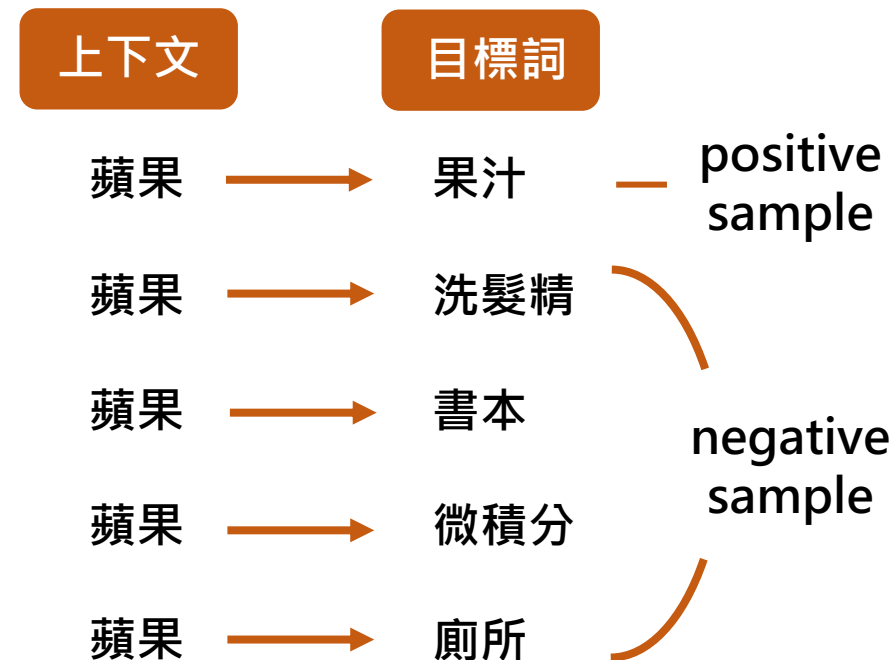


常用參數名稱	值/方法
加快訓練速度的方法(hs)	0=negative sampling

結合**負採樣**(negative sampling)

- 加快模型訓練的速度
- 訓練模型效果更好

舉例：「好想喝蘋果果汁喔。」



TF-IDF 公式

使用 **TF-IDF** 演算法對詞進行加權平均。

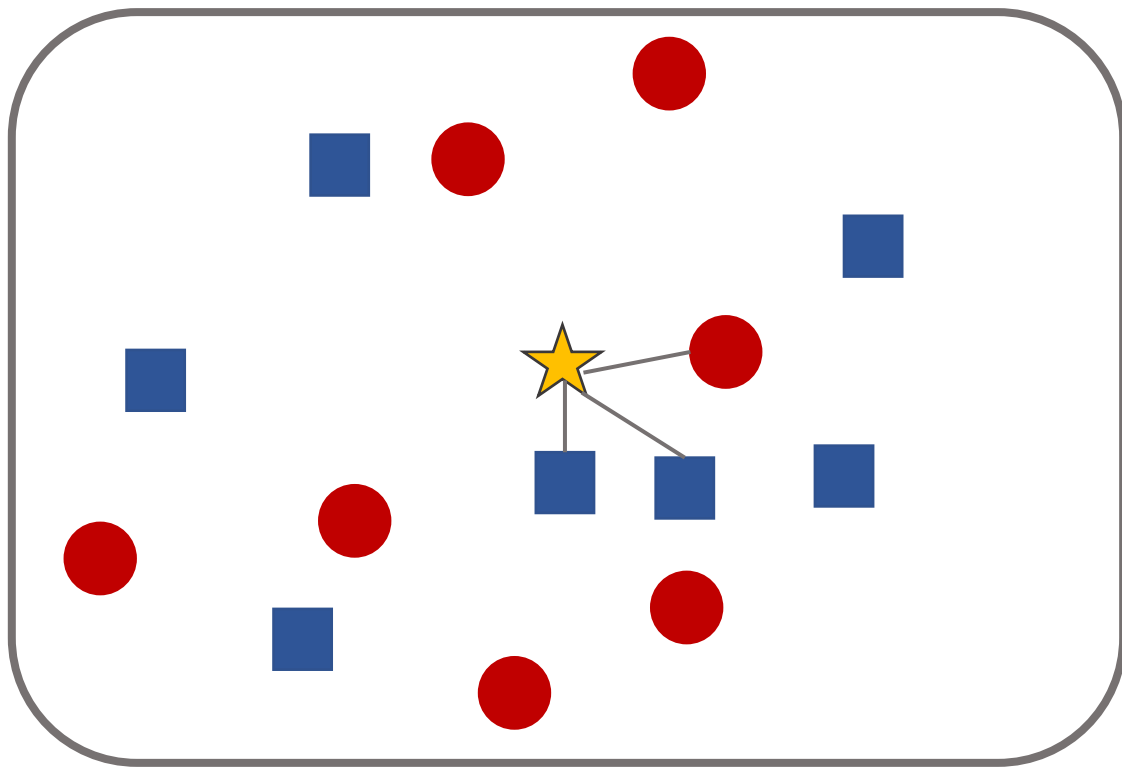
- TF：詞在一份文件出現的次數，即為詞頻
- IDF：包含詞的文件比例

TF×IDF值越大，給予該詞彙較大**權重**

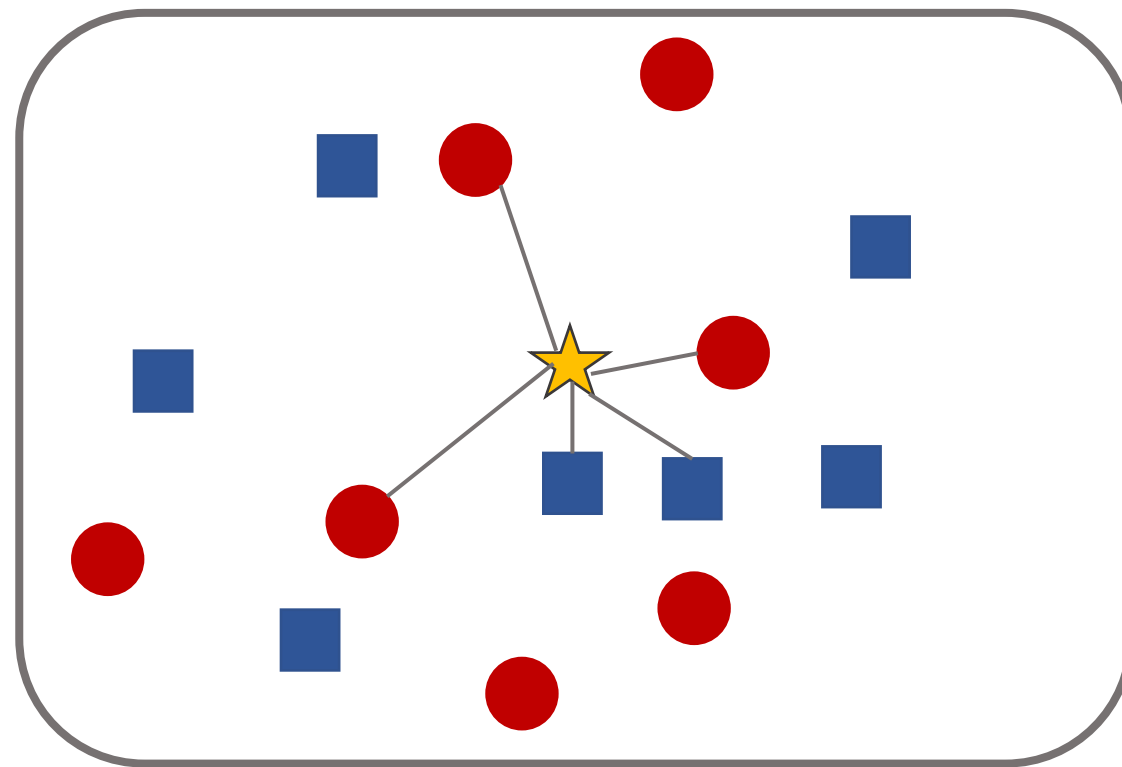
$$score_{t,d} = \underbrace{\frac{n_{t,d}}{\sum_{k=1}^T n_{k,d}}}_{\text{TF 值}} \times \underbrace{\log\left(\frac{D}{d_t}\right)}_{\text{IDF 值}}$$



KNN模型



K=3 → ■



K=5 → ●

SVM公式

- Gaussian Radial Basis Function kernel (RBF)
將資料投射到無限維

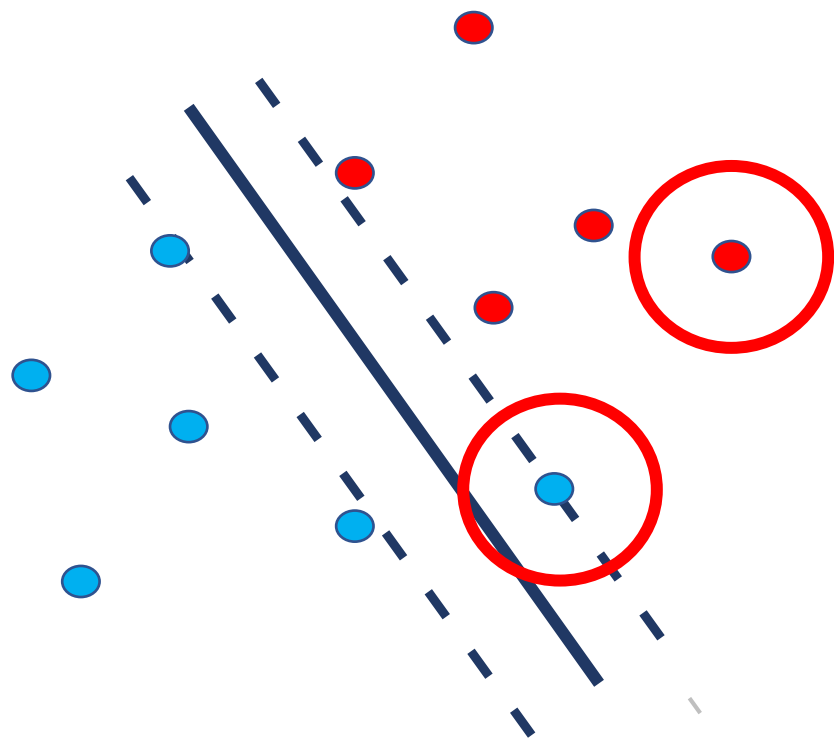
$$k(x, y) = e^{-\gamma \|a-b\|^2}$$

預測結果為資料與決策邊界的距離，透過 Platt Scaling
將結果投射到 $(0, 1)$ 之間，使其成為機率值

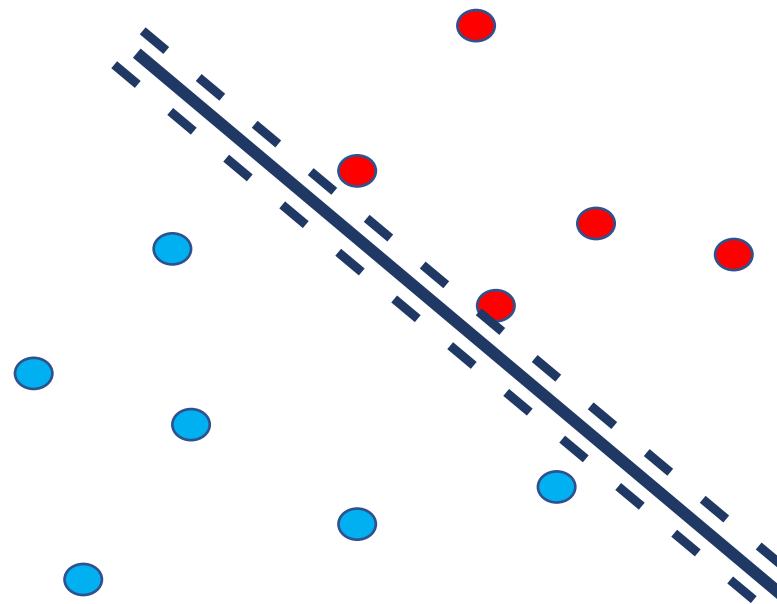
$$P(y_i = 1|f_i) = \frac{1}{1+e^{Af_i+B}}$$



SVM模型 哪一條線比較合適?



soft margin



hard margin

貝式分類器

假設所有特徵皆為獨立，透過貝式定理，計算在已知資料下哪個目標發生的機率最大。

- 高斯貝式分類器 (GaussianNB)
用於特徵 x 為連續變數且符合常態分佈，將連續數值離散化

$$P(x = v|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}$$



貝式分類器

- 多項式貝式分類器

用於特徵 x 為類別變數時，經常用於文本分析模型

$$P(Y_k) = \frac{N_{Y_k} + \alpha}{N + k_\alpha}$$

$$P(X_i|Y_k) = \frac{N_{Y_k, x_i} + \alpha}{N_{Y_k} + n\alpha}$$



Naïve Bayes 模型

舉例：

「快點過來！」

10個句子

4句 😊

4句有3句「快點」

6句 😡

6句有2句「快點」

$$P(\text{😊}) \cdot P(\text{快點} \mid \text{😊}) = \frac{4}{10} \times \frac{3}{4} = \frac{3}{10}$$

$$P(\text{😡}) \cdot P(\text{快點} \mid \text{😡}) = \frac{6}{10} \times \frac{2}{6} = \frac{2}{10}$$

Naïve Bayes 模型

舉例：



「快點過來！」

10個句子

4句 😊

4句有1句「過來」

6句 😡

6句有3句「過來」

$$P(\text{😊}) \cdot P(\text{快點} \mid \text{😊}) \cdot P(\text{過來} \mid \text{😊}) = \frac{4}{10} \times \frac{3}{4} \times \frac{1}{4} = \frac{3}{40}$$

$$P(\text{😡}) \cdot P(\text{快點} \mid \text{😡}) \cdot P(\text{快點} \mid \text{😡}) = \frac{6}{10} \times \frac{2}{6} \times \frac{3}{6} = \frac{4}{40}$$