Tweets Analysis –
whether Tweets indicate disaster or not

Statistical Computing Project

指導教授：鄭又仁 教授

學生：蘇柏庄、吳岱錡、陳威宇、王彥翔、蔡雅欣

Contents

1 Introduction and Data Description

1.1 分析目的

現今 Twitter 成爲了人們經常使用的資訊交流平臺，意外或事故的情報分享也不例外，人手一機的條件更讓大家幾乎能即時的發佈自己遇到的緊急事態，然而也會存在一些包含事故字眼的貼文，實際上只是作者幽默的玩笑。因此我們便想去瞭解，是否能藉由統計方法來判別 Twitter 上的事故貼文是否爲真。此外我們也會試著去比較幾種演算法的差異和判斷正確率。希望透過這次其中報告，將這半學期所學融會溝通。

1.2 資料說明

為探究是否一則推特貼文是關於一個真實發生的事故，我們取用 appen(https://appen.com/resources/datasets) 上的公開資料庫。該資料集上有 7613 個觀察值，包含 4 個欄位：

變數類別	變數名稱	變數解釋
Covariates	ID	推特貼文識別碼
	Text	推特貼文文字內容
	Location	推特的發文地點
	KeyWord	推特貼文中的特定單詞
	Target	推特中的事故是否真實發生

其中推特中的事故真實發生的有 3271 筆資料，沒有真實發生的有 4342 筆資料。

1.2.1 資料缺失值

在 7613 個觀察值，關於特定單詞 (KeyWord) 有 61 個缺失值、推文發生地點 (Location) 有 2533 個缺失值，其中推文發生地點 (Location) 的缺失值占比高達 33%。由於地點資料無法應用其他方式生成，在以下分析中將不予以採用。

```
##        id  keyword location      text   target
##         0       61     2533         0        0
```

1.2.2 新增特徵資料

在本份資料中，將新增推特貼文文字內容長度資料 (TextLength)，供之後在推特中的事故是否真實發生的兩組中，分析貼文長短是否有顯著不同。

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0    78.0   108.0   101.4   134.0   157.0
```

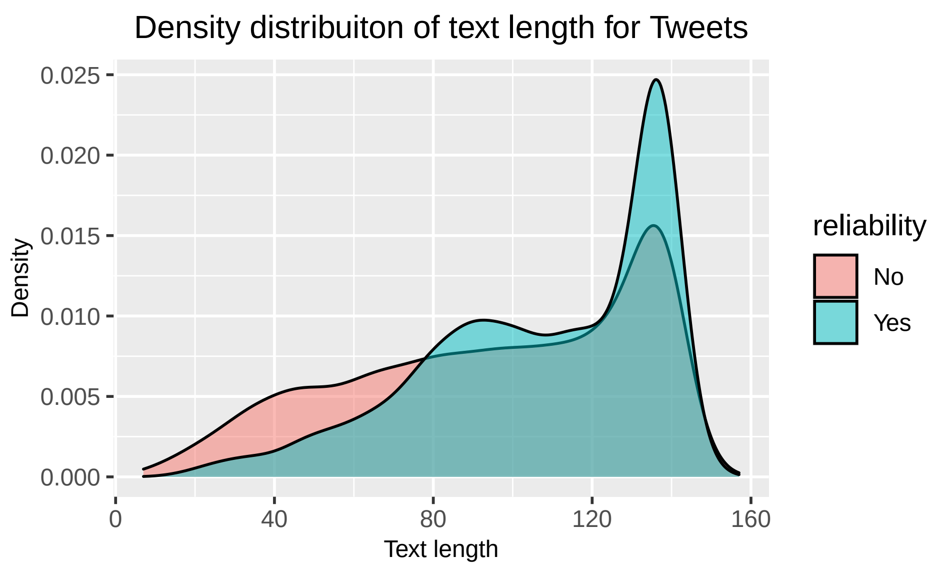
1.3 分析流程

在本次的文字探勘當中，我們的目標為將其分為兩個叢集，第一步我們會進行資料探索，了解關鍵字與全文內容、發文地點等的關係。對資料有一定的認識後，我們使用了 3 種機器學習的演算法建立模型對資料集進行集群分析，並藉由模擬資料來檢定各個模型的準確率。

2 EDA

2.1 Text Features Analysis

- Analysis on text length



事故實際未發生時，其分布會較為平均；但倘若實際發生的話，其推文文長會集中在字數較長的區間，而在字數較少的區間其頻率皆低於未發生的情況。

- **Two Sample T-test**(雙樣本平均數差異 t-檢定)

首先：$x-->$ 真實事故推文字數，$y-->$ 錯誤警報推文字數

$t-test$ 結果得出 $p-value < 2.2e-16$ 極小，故判斷 x 與 y 間「推文字數」差異顯著；其 95% 信賴區間為 $(10.94068, 13.87252)$，且 x 平均值為 108.113，y 平均值為 95.707。

2.2 Text of Tweets Analysis

2.2.1 Text Cleansing

- **Unigrams**

1. Convert text to lower case
2. Remove URLs, Usernames, numbers, punctuation, extra whitespaces, etc. (remove whitespaces after step 4.)
3. Remove stopwords (English), common words.
4. Stemming words to root words

所謂的 Unigram 為一元分詞，將文本中的字詞分成單詞來分析。

- **Bigrams**

1. Convert text to lower case
2. Remove numbers, punctuation, etc. (remove whitespaces after step 4.)
3. Remove stopwords (English).
4. Ignore overly sparse and common terms (less than 1%, more than 80%)

所謂的 Bigram 為二元分詞，將文本中的字詞分成雙詞來分析；而對於雙詞分析，將不對字詞做詞根抽取 (Stemming)，以保存雙詞完整意思。

2.2.2 Analysis on Tweets - WordCloud

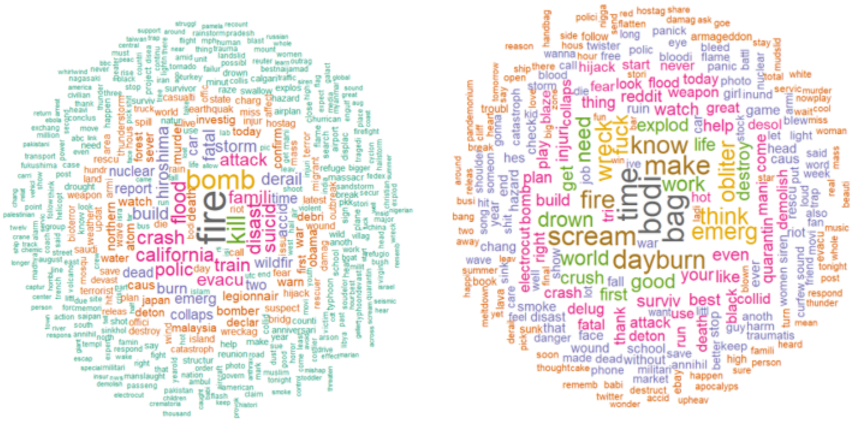


Figure 1: WordCloud of Disaster Text versus Non-disaster Text

左圖為真實事故的文字雲，右圖為錯誤警報的文字雲。明顯地，左圖出現次數高的詞：「**fire**」、「**bomb**」、「**kill**」、「**attack**」、「**crash**」、「**disaster**」等，其與人員傷亡必會有一定程度的連結；而右圖大多為無關緊要的文字或是使用上較為聳動的用詞。

2.2.3 Analysis on Tweets - Unigrams

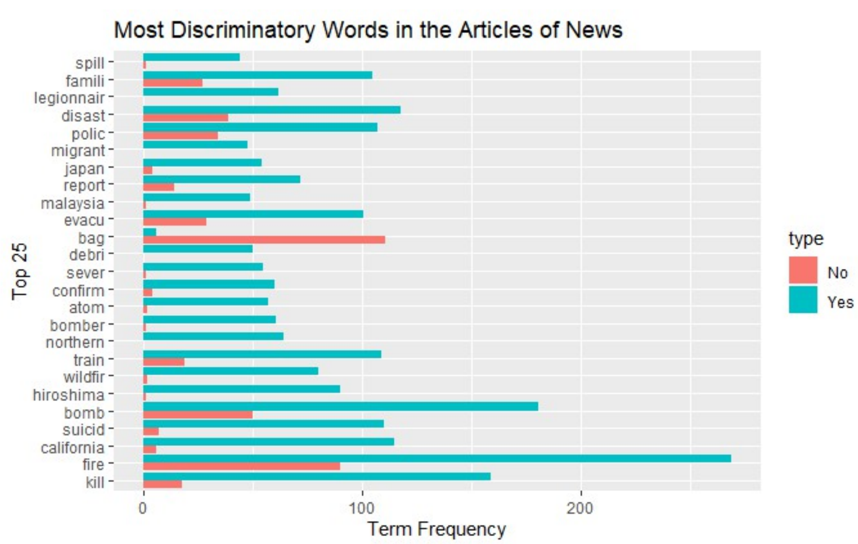


Figure 2: Unigram Top Discriminatory Words

- **Bomb**(炸彈)：「有事故」的出現頻率遠高於「實際未發生」。
- **Fire**：「有事故」的出現頻率遠高於「實際未發生」。

- **Kill**：「有事故」的出現頻率遠高於「實際未發生」。

這三者為描述上較嚴肅的詞，且與人身安全密切相關，故推文者理應會較為慎重地使用，其推文的真實性的機會相對較高。

2.2.4 Analysis on Tweets - Bigrams

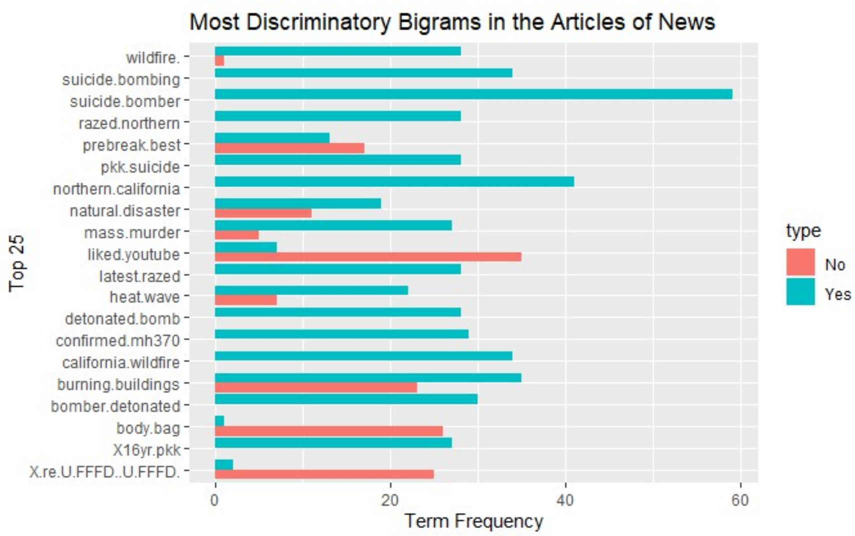


Figure 3: Bigram Top Discriminatory Words

根據雙詞分析：

當「**suicide**（自殺）」與「**bomber**（炸彈客）」、「**bombing**（轟炸）」同時出現時，不僅頻率高且在資料集中皆為「真實事故」；而由於當地在資料收集的時候剛好遭逢大火，故「**northern california**（北加州）」、「**california wildfire**（加州野火）」同時出現時不僅必為真實事件且其出現頻率亦高。

雙詞如「**liked youtube**」同時出現時高機率為「未發生事故」，故判斷內文有可能為騙取 youtube 點擊率而捏造的假消息；又雙詞如「**burning buidings**」同時出現時真假「近乎參半」，因此推斷可能是頻繁發生的社會案件且時不時為錯誤警報。

3 Method

3.1 tf-idf

tf-idf (term frequency–inverse document frequency) 是一種用於文件探勘的字詞權重方法，用以評估詞（Term）對於文件（Document）的重要程度，於文獻中也證實能於 Text Classification 的模型，作為 Covariates 使用。

- $tf_{t,d}$ (term frequency，詞頻) 衡量各個詞於各個文件中出現的頻率，並將其標準化。

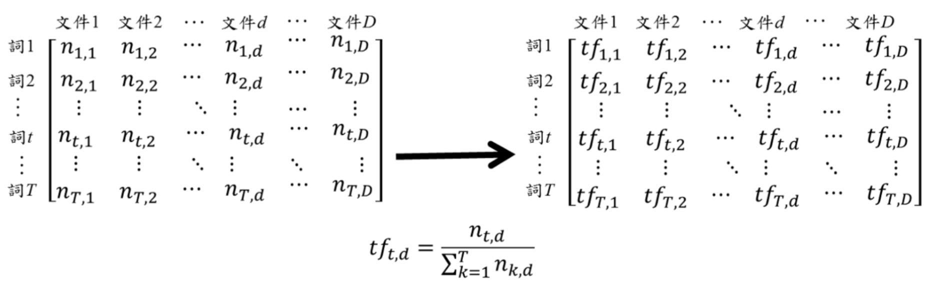


Figure 4: tf - term frequency

- idf_t (inverse document frequency，逆向文件頻率) 則計算每個詞於整份文件中的重要性，若其於每份文件皆曾出現，則較為不重要。

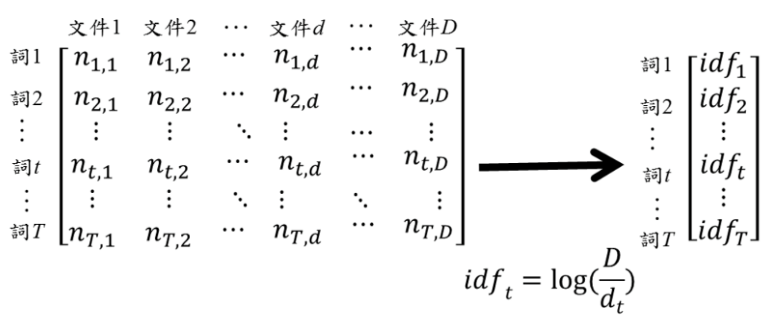


Figure 5: idf - inverse document frequency

- tf-idf 由 $score_{t,d}$ 來計算各個詞於每個文件中的重要性權重，將前兩者所計算的 idf_t 以及 $tf_{t,d}$ 進行矩陣相乘後，可得其值。

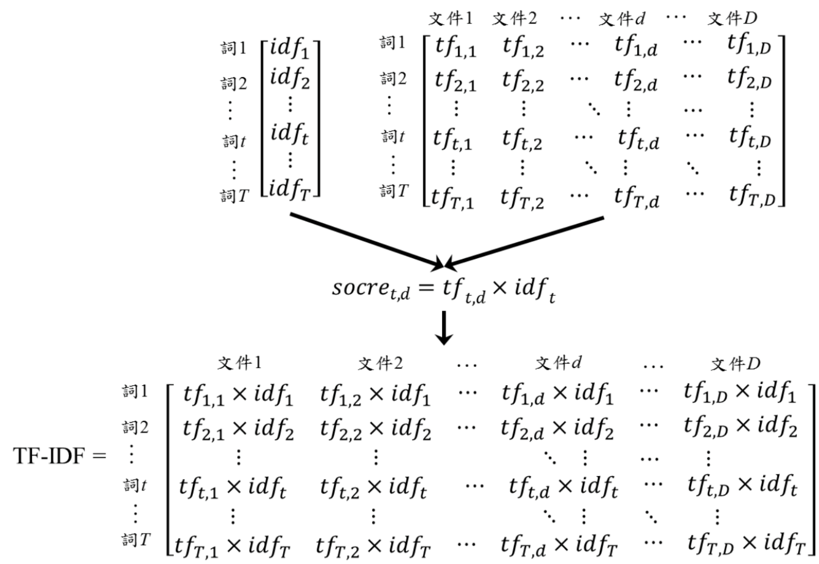


Figure 6: tf-idf

3.2 K-Means

K-means 是一個簡單易懂且利於解釋的分類演算法, 其想法為在事先給定 K 個群集下，最小化群內的資料與群心的誤差平方和，公式可以寫爲

$$\underset{\mu}{argmin} \sum_{c=1}^{K} \sum_{i=1}^{n_c} ||x_i - \mu_c||^2 \Big|_{x_i \in S_c}$$

μ_c 就是群心，$||x-y||$ 就是算歐氏距離 (Euclidean distance)，S_c 則代表第 c 個群集 (cluster)。其演算法為：

1. 設定 k 個群心

$$\mu_c^{(0)} \in R^d, \quad c = 1, 2, ..., K$$

2. 將每個樣本分到與其最接近的群心所屬的群集中

$$S_c^{(t)} = \{x_i : ||x_i - \mu_c^{(t)}|| \leq ||x_i - \mu_{c^*}^{(t)}||, \forall\ i = 1, ..., n\}$$

3. 結合新加入的資料計算新的群心 (第 c 群內有 n_c 個資料)

$$\mu_c^{t+1} = \frac{sum(S_c^{(t)})}{n_c} = \sum_{i=1}^{n_c} x_i \Big|_{x_i \in S_c}$$

4. 重複 2 和 3 直到群心收斂不變

$$S_c^{(t+1)} = S_c^{(t)}, \quad \forall\ c = 1, ..., K$$

群集數量 K 值的決定與起始群心的決定都會對模型造成影響，群集數量可以使用 Hierarchical Clustering 的方法尋找適合的數值，而起始值的問題則可藉由以不同的起始值多次執行模型，再從中取出最佳結果的方式來排除。

3.3 Gaussian Mixture Model

高斯混合模型（Gaussian Mixture Model，簡稱 GMM) 是單一高斯機率密度函數的延伸，指的是多個高斯分布函數的線性組合，由於 GMM 能夠平滑地近似任意形狀的密度分佈，因此近來常被用在語音辨識並得到不錯的效果。

- 單一高斯機率密度函數

$$N(x; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}} exp\Big[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\Big]$$

其中 μ 代表此密度函數的中心點，Σ 則代表此密度函數的共變異矩陣（Covariance Matrix)。

- 推導 GMM 流程：

設一隨機變數：

$$\mathbf{z} = [z_1, z_2, ..., z_K]$$

當 $z_k = 1$ 且 $z_i = 0,\ i \neq k$，代表第 k 個袋子被選中，其屬於第 k 個叢集；由於此變數無法量測，故稱之為隱含變量（latent variable)。

$$p(z_k = 1) = \alpha_k, \quad k = 1, ..., K$$

其中 α_k 為屬於第 k 個叢集的事前機率。

概似函數：

$$p(x|\mathbf{z}) = \prod_{k=1}^{K} p(x|z_k = 1)^{z_k} = \prod_{k=1}^{K} p(x|\mu_k, \Sigma_k)^{z_k}$$

把所有 z 可能的概似函數乘上先驗機率再相加：

$$p(x|\Theta) = \sum_{z} p(x|\mathbf{z})p(z) = \sum_{k=1}^{K} p(x|z_k = 1)p(z_k = 1) = \sum_{k=1}^{K} \alpha_k p(x|\mu_k, \Sigma_k)$$

其中的 Θ 稱為高斯混合模型的參數集。

$$\Theta = \{\alpha_k; \mu_k, \Sigma_k\}_{k=1}^{K}$$

把高斯分佈寫進去替換以上機率模式，得到以下高斯混合模型：

$$\begin{aligned} p(x|\Theta) &= \sum_{k=1}^{K} \alpha_k N(x|\mu_k, \Sigma_k) \\ \sum_{k=1}^{K} \alpha_k &= \ 1, \quad 0 \le \alpha_k \le 1 \\ N(x|\mu_k, \Sigma_k) &= \frac{1}{\sqrt{(2\pi)^d |\Sigma_k|}} exp\Big[-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)\Big] \end{aligned}$$

3.3.1 EM Algorithm

給定樣本集合 $\{x_1, x_2, ..., x_n\}$，在選取 n 個樣本後，其概似函數為：

$$\begin{aligned} L(\Theta) &= \prod_{i=1}^{n} p(x_i|\Theta) \\ \Rightarrow lnL(\Theta) &= \sum_{i=1}^{n} lnp(x_i|\Theta) \\ &= \sum_{i=1}^{n} ln\Big\{\sum_{k=1}^{K} p(x_i|z_k = 1)p(z_k = 1)\Big\} \\ &= \sum_{i=1}^{n} ln\Big\{\sum_{k=1}^{K} \alpha_k N(x_i|\mu_k, \Sigma_k)\Big\} \end{aligned}$$

EM 是一種不斷反覆運算的演算法，所以參數會不斷的更新，此處假設第 t 與 $t+1$ 次估計的參數如下：

$$\Theta^{(t)} = \Big\{\alpha_k^{(t)};\, \mu_k^{(t)},\, \Sigma_k^{(t)}\Big\}_{k=1}^{K}$$

$$\Theta^{(t+1)} = \Big\{\alpha_k^{(t+1)};\, \mu_k^{(t+1)},\, \Sigma_k^{(t+1)}\Big\}_{k=1}^{K}$$

- 演算法流程

假設給定樣本集合 $x_1, x_2, ..., x_n$，

1. 初始化參數：設定 K 個數，t(第 t 次計算) 設定為 0

$$\Theta^{(0)} = \left\{\alpha_k^{(0)};\, \mu_k^{(0)},\, \Sigma_k^{(0)}\right\}_{k=1}^{K}$$

2. **E-Step**

假設所有參數 $\Theta^{(t)}$ 已知，計算下式：

$$w_k^{(t)}(x_i) = p(z_k = 1|x_i) = \frac{\alpha_k^{(t)} N\left(x_i|\mu_k^{(t)}, \Sigma_k^{(t)}\right)}{\sum_{j=1}^{K} N\left(x_i|\mu_j^{(t)}, \Sigma_j^{(t)}\right)},\ \forall i, k$$

後驗機率 $p(\mathbf{z}|x)$ 如下：

$$w_k(x) = p(z_k = 1|x) = \frac{p(x|z_k = 1)p(z_k = 1)}{p(x)} = \frac{\alpha_k N(x|\mu_k, \Sigma_k)}{\sum_{i=1}^{K} N(x|\mu_i, \Sigma_i)}$$

其中機率和為 1:

$$\sum_{i=1}^{K} w_k(x) = \sum_{i=1}^{K} p(z_k = 1|x) = 1$$

藉此方便利用 EM 演算法來估計 GMM 的參數。

3. **M-Step**

利用 MLE 去估計 $q(\Theta^{(t)}, \Theta^{(t+1)})$ 的參數 $\Theta^{(t+1)}$

$$\Theta^{(t+1)} = \left\{\alpha_k^{(t+1)};\, \mu_k^{(t+1)},\, \Sigma_k^{(t+1)}\right\}_{k=1}^{K}$$

參數 $\mu_k^{(t+1)}, \Sigma_k^{(t+1)}, \alpha_k^{(t+1)}$ 的解如下：

$$\begin{aligned}
\mu_k^{(t+1)} &= \frac{1}{n_k^{(t)}} \sum_{i=1}^{n} w_k^{(t)}(x_i) x^{(i)} \\
\Sigma_k^{(t+1)} &= \frac{1}{n_k^{(t)}} \sum_{i=1}^{n} \sum_{k=1}^{K} w_K^{(t)}(x_i) \left(x^{(i)} - \mu_k^{(t+1)}\right)\left(x^{(i)} - \mu_k^{(t+1)}\right)^T \\
\alpha_k^{(t+1)} &= \frac{n_k^{(t)}}{n}
\end{aligned}$$

其中 $n_k^{(t)} = \sum_{i=1}^{n} w_k^{(t)}(x_i)$。

4. 重複 2~3，直到滿足收斂條件 (參數收斂或是概似函數收斂)

3.4 Logistic

邏輯迴歸 (Logistic Regression)，是用於處理二元變數 (binary response) Y 及解釋變數 (explanatory variable) X 間的關係的統計方法。其中 X 可為分類型變數，也可以是連續型變數。首先，定義 $\pi(x) = P(Y = 1 \ X = x)$ 模型爲

$$\pi(x) = \frac{exp(\alpha + \beta x)}{1 + exp(\alpha + \beta x)}$$

其回傳為介於 0~1 之間的數值。

而邏輯迴歸便建立在 $\frac{\pi(x)}{1-\pi(x)}$ 這樣的成功幾率和失敗機率的比值上，我們稱之為 **Odd**。在對其取對數後，便是命名為 logit 的計算法，其同時具備著線性結構。

$$Logit[\pi(x)] = log\frac{\pi(x)}{1 - \pi(x)} = \alpha + \beta x$$

但有多個解釋變數時，如 $x = (x_1, x_2, ..., x_p)$，其模型可以寫為

$$Logit[\pi(x)] = \alpha + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p$$

回推可知其

$$\pi(x) = \frac{exp(\alpha + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p)}{1 + exp(\alpha + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p)}$$

在估計參數 α 與 β_j 時，我們使用最大蓋似函數估計 (MLE)，首先假設有 n 個樣本，且 $\pi(x_i)$ 為伯努利分配 (Bernoulli distribution)，其 pdf 為 $\pi(x_i)^{y_i}[1 - \pi(x_i)^{1-y_i}]$，此時蓋似函數為

$$\begin{aligned}
\Pi_{i=1}^{n}\pi(x_i)^{y_i}[1 - \pi(x_i)^{1-y_i}] &= \Pi_{i=1}^{n}\Big\{exp\Big(log(\frac{\pi(x)}{1-\pi(x)})^{y_i}[1 - \pi(x)]\Big)\Big\} \\
&= \Pi_{i=1}^{n}\Big\{exp\Big(log(y_i\frac{\pi(x)}{1-\pi(x)}\Big)\Big\}\Big\{\Pi_{i=1}^{n}[1 - \pi(x)]\Big\} \\
&= \Big\{exp[\sum_{i=1}^{n} y_i log\frac{\pi(x)}{1-\pi(x)}]\Big\}\Big\{\Pi_{i=1}^{n}[1 - \pi(x)]\Big\}
\end{aligned}$$

由於 $log\frac{\pi(x)}{1-\pi(x)} = \sum_j \beta_j x_{ij}$，因此 $exp[\sum_{i=1}^{n} y_i log\frac{\pi(x)}{1-\pi(x)}]$，便可寫作 $exp\Big[\sum_j(\sum_i y_i x_i j)\beta_j\Big]$，同時 $[1-\pi(x)] = 1 + exp\big(\sum_j \beta_j x_{ij}\big)^{-1}$，此時蓋似函數便可寫做

$$L(\beta) = \sum_j (\sum_i y_i x_{ij})\beta_j \ - \ \sum_i log[1 + exp(\sum_j \beta_j x_{ij})]$$

接著找出最大蓋似函數便是找到 $L(\beta)$ 極值的一階條件

$$\frac{\partial L(\beta)}{\partial \beta_j} = \sum_i y_i x_{ij} \ - \ \sum_i x_{ij} \frac{exp(\sum_k \beta_k x_{ik})}{1 + exp(\sum_k \beta_k x_{ik})} = 0$$

其解為

$$\sum_i y_i x_{ij} \ - \ \sum_i \hat{\pi}_i x_{ij} = 0 \hat{\pi}_i = \frac{exp(\sum_k \hat{\beta}_k x_{ik})}{1 + exp(\sum_k \hat{\beta}_k x_{ik})}$$

$\hat{\pi}_i$ 同時是 $\pi(x_i)$ 的 MLE，但由於沒有封閉解，故需要用牛頓法等數值方法迭代求出 $\hat{\beta}_k$，進而得到目標函數 $\hat{\pi}_i$。

3.5 Random Forest

Random Forest 是一種 Ensemble Learning 的演算法，而 Ensemble Learning 概念上是結合多個弱學習器來建構一個強穩的模型。每次會用類似 Bootstrap Aggregation 的方法取得一個新的 Decision Tree，再將所有的 Decision Tree 結合起來。

其中 Decision Tree 是一種監督式機器學習模型，利用像樹一樣的圖形去建構預測模型，適用於類別及連續資料類型的預測，相較於其他 Machine Learning 的模型，Decision Tree 的過程直覺、單純且執行效率也很高。Decision Tree 的特點是每個決策階段都很清楚，每個節點代表一個屬性（變數）、每個分支代表對應該屬性的某些可能值（變數範圍）、每個葉節點代表滿足對應路徑的條件下之最終的預測值。

因此將所有的 Decision Tree 結合起來不僅能夠保持 Decision Tree 的差異性，還能減少 fully-grown 造成的 overfitting。此外 Random Forest 最大的精神就是隨機，除了樣本是利用 Bootstrap 採取抽後放回的隨機抽取概念外，變數也是採取隨機抽取的方式，也因此讓 Random Forest 具有高度多樣性。

$$Model: \ \hat{f}^{rf}(x) = argmax \sum_{b=1}^{B} I\left(\hat{f}^{tree,b}(x) = k\right)$$

Random Forest 發展出了一套獨特的 Validation 方法，由於樣本利用 Bootstrap 方法抽取後，會導致有一部份的資料沒有被取用，因此 Random Forest 將這些沒有被抽取到的資料

當作類似於測試集的形式，稱之為 Out-of-Bag Data，並可以運用這筆資料計算 Out-of-Bag Error。而透過 Out-of-Bag Error，我們可以運用這個值去選取模型最適合的參數。以下為參數調控：

初始參數給定 $ntree = 211$ (透過 Out-of-Bag Error 選定)，主要運用 grid search 的方法去調整參數，概念就是針對每一個模型參數組合，都會配適出一個模型，然後從中挑選表現最佳的模型。這次我們主要調控的參數有：mtry、nodesize、sampsize。

模型參數	參數解釋
ntree	numbers of tree，我們希望有足夠的決策樹來穩定模型的誤差，但過多的樹會導致模型沒有效率。
mtry	每次在決定切割變數時，我們選擇要進行隨機抽樣的變數個數。
nodesize	指定決策樹節點的最小個數，默認情況下，判別模型為 1，回歸模型為 5。
sampsize	訓練每棵樹的樣本隨機採樣比例，較低的值使得算法更加保守，防止 overfitting，但是太小的值也會造成 underfitting。

此外 Random Forest 的結果可以計算每個特徵的重要程度，在 R 語言中，最後估計出的模型會提供「Mean Decrease Accurary」及「Mean Decrease Gini」，兩者皆可用來進行特徵選取。Mean Decrease Accurary 大致上是將 data 中第 i 個變數抽取出後隨機打亂再放入 data 中，將新 data 代入模型計算出新的 Accuracy 後，比較 Accuracy 與原先的差異。Mean Decrease Gini 則是計算該變數讓整個模型之不純度下降的比例。

其中不純度（impurity）可以為

- Misclassification rate

$$\phi(p_1, p_2, ..., p_k \,|\, t) = 1 - max(p_1, p_2, ..., p_k|t)$$

- Entropy

$$\phi(p_1, p_2, ..., p_k \,|\, t) = -\sum_{i=1}^{K} p(j \,|\, t) \; log \; p(j \,|\, t)$$

- Gini Index

$$\phi(p_1, p_2, ..., p_k \,|\, t) = 1 - \sum_{i=1}^{K} p(j \,|\, t)^2$$

其中 $p(k \mid t)$ 由 $\hat{p}(k \mid t) = \frac{1}{n_t} \sum_{x_i \in node_t} I(y_i = k)$ 估計而得。

3.6 Latent Dirichlet Allocation

Latent Dirichlet Allocation 是一種主題模型，其可將文檔中每篇文檔的主題以機率分布的形式給出，從而透過分析文檔資料抽取其主題分布，進一步根據主題分布進行聚類或文本分類。

在 LDA 模型中，以生成文檔的角度來解析：

1. 先按照先驗機率從 $P(d_m)$ 抽取一篇文檔 d_m
2. 從 $Dirichlet(\alpha)$ 中抽樣生成文檔 d_m 的主題分布 θ_m
3. 從主題的多項式分布 θ_m 中抽樣生成文檔 d_m 第 n 個詞的主題 $z_{m,n}$
4. 從 $Dirichlet(\beta)$ 中抽樣生成主題 $z_{m,n}$ (k) 對應的詞語分布 $\phi_{z_{m,n}}$ (ϕ_k)
5. 從詞語的多項式分布 $\phi_{z_{m,n}}$ 中採樣最終生成詞語 $w_{m,n}$

其中，Dirichlet 分布是多項式分布的共軛先驗機率分布，其可表示成如下：

$$Dirichlet(\vec{p} \mid \vec{\alpha}) + MultiCount(\vec{m}) = Dirichlet(\vec{p} \mid \vec{\alpha} + \vec{m})$$

此外，LDA 的模型結構圖如下：

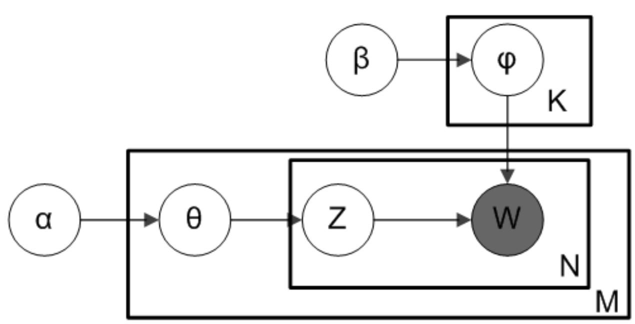


Figure 7: Bayesian Network of Latent Dirichlet Allocation

上圖中，灰色圓圈為觀察到的變量，也就是文字，而其他白色圓圈的皆為隱變量或是參數；其中 θ 代表主題分布，ϕ 代表詞語分布，α 是主題分布 θ 的 Dirichlet 先驗分布的參數，β 是詞語分布 ϕ 的先驗分布的參數，N 代表文檔中的單詞總數，M 代表文檔的總數。

文章由 LDA 上述過程生成過程：

假設語庫共有 M 篇文章，每篇文章下的主題分佈是一個從 $Dirichlet(\alpha)$ 先驗分佈中採樣得到的多項式分佈 θ，每個主題下的詞語分佈是一個從 $Dirichlet(\beta)$ 先驗分佈中採樣得到的多項式分佈分佈 ϕ。對於某一文章中的第 n 個詞語，首先從該文章中出現的每個主題的多項式分布 (主題分布θ) 中採樣一個主題，接著在此主題所對應的多項式分布 (詞語分布ϕ) 中採樣一個詞語。

重複上述隨機生成過程，直到所有 M 篇文章完成；而此 M 篇生成文章會對應於 M 個獨立的 $Dirichlet-Multinomial$ 共軛結構、K 個主題將會對應於 K 個獨立的 $Dirichlet-Multinomial$ 共軛結構。其結構如下：

- 生成文檔中的所有詞語對應的主題的 $Dirichlet-Multinomial$ 共軛結構：

$$\vec{\alpha}\underbrace{\xrightarrow{\hspace{6em}}}_{Dirichlet}\vec{\theta}_m\underbrace{\xrightarrow{\hspace{6em}}}_{Multinomial}\vec{z}_m$$

- 生成文檔中的每個主題所對應的詞語的 $Dirichlet-Multinomial$ 共軛結構：

$$\vec{\beta}\underbrace{\xrightarrow{\hspace{6em}}}_{Dirichlet}\vec{\varphi}_k\underbrace{\xrightarrow{\hspace{6em}}}_{Multinomial}\vec{w}_{(k)}$$

- 參數估計 θ and ϕ：

$$\begin{array}{llll}
\vec{\theta}_m \sim Dir(\vec{\theta}_m|\vec{\alpha}) & -(1) & z_{m,n} \sim Multi(z_{m,n}|\vec{\theta}_m) & -(2) \\
\vec{\phi}_k \sim Dir(\vec{\phi}_k|\vec{\beta}) & -(3) & w_{m,n} \sim Multi(w_{m,n}|\vec{\phi}_k, z_{m,n}) & -(4)
\end{array}$$

- 聯合分布

$$p(\vec{z},\vec{w}|\vec{\alpha},\vec{\beta}) = p(\vec{w}|\vec{z},\vec{\beta})p(\vec{z}|\vec{\alpha})$$

Eq.(1) 與 Eq.(2) 聯合式 Dirichlet-Multinomial Conjugate Distribution，其中多項式分配為：

$$p(\vec{z}|\Theta) = \prod_{m=1}^{M} \prod_{k=1}^{K} \vartheta_{m,k}^{n_m^{(k)}}$$

Eq.(3) 與 Eq.(4) 聯合式 Dirichlet-Multinomial Conjugate Distribution，其中多項式分配為：

$$p(\vec{w}|\vec{z},\phi) = \prod_{k=1}^{K} \prod_{t=1}^{V} \varphi_{k,t}^{n_k^{(t)}}$$

Dirichlet-Multinomial Models:

- 第一項因子 $p(\vec{w}|\vec{z},\vec{\beta})$：根據確定主題 $\vec{z}$ 與詞語分布的先驗分布參數 β 採樣詞語

$$\begin{aligned} p(\vec{w}|\vec{z},\vec{\beta}) &= \int p(\vec{w}|\vec{z},\phi)p(\phi|\vec{\beta})\, \mathrm{d}\phi \\ &= \int \prod_{k=1}^{K} \prod_{t=1}^{V} \varphi_{k,t}^{n_k^{(t)}} \frac{1}{\Delta(\vec{\beta})} \prod_{k=1}^{K} \prod_{t=1}^{V} \varphi_{k,t}^{\beta_k - 1}\, \mathrm{d}\vec{\phi} \\ &= \int \prod_{k=1}^{K} \frac{1}{\Delta(\vec{\beta})} \prod_{t=1}^{V} \varphi_{k,t}^{n_k^{(t)}+\beta_k-1} \mathrm{d}\vec{\phi} \\ &= \prod_{k=1}^{K} \frac{\Delta(\vec{n}_k + \vec{\beta})}{\Delta(\vec{\beta})}, \quad \vec{n}_k = \{n_k^{(t)}\}_{t=1}^{V} \end{aligned}$$

其中，$\vec{n}_k$ 為構成主題 k 的詞語項數向量。

- 第二項因子 $p(\vec{z}|\vec{\alpha})$：根據主題分布的先驗分布參數 α 採樣主題

$$\begin{aligned} p(\vec{z}|\vec{\alpha}) &= \int p(\vec{z}|\Theta)\, p(\Theta|\vec{\alpha})\, \mathrm{d}\Theta \\ &= \int \prod_{m=1}^{M} \prod_{k=1}^{K} \vartheta_{m,k}^{n_m^{(k)}} \frac{1}{\Delta(\vec{\alpha})} \prod_{m=1}^{M} \prod_{k=1}^{K} \vartheta_{m,k}^{\alpha_k - 1}\, \mathrm{d}\vec{\theta}_m \\ &= \int \prod_{m=1}^{M} \frac{1}{\Delta(\vec{\alpha})} \prod_{k=1}^{K} \vartheta_{m,k}^{n_m^{(k)}+\alpha_k-1} \mathrm{d}\vec{\theta}_m \\ &= \prod_{m=1}^{M} \frac{\Delta(\vec{n}_m + \vec{\alpha})}{\Delta(\vec{\alpha})}, \quad \vec{n}_m = \{n_m^{(k)}\}_{k=1}^{K} \end{aligned}$$

其中，$\vec{n}_m$ 為構成文檔 m 的主題數向量。

並且兩個因子當中的 $\Delta(\vec{\alpha})$ 稱為 Dirichlet 分布的歸一化係數：

$$\Delta(\vec{\alpha}) = \frac{\prod_{k=1}^{dim(\vec{\alpha})} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{dim(\vec{\alpha})} \alpha_k)} = \int \prod_{k=1}^{dim(\vec{\alpha})} p_k^{\alpha_k - 1} \, \mathrm{d}\vec{p}$$

有了聯合分布 $p(\vec{w}, \vec{z})$ 後，便可以通過其計算在給定觀測變量 $\vec{w}$ 下的隱變量 (latent variable)z 的後驗分布 $p(\vec{z}|\vec{w})$ 來進行貝斯分析 (Bayesian Analysis)。接著將求解第 m 篇文檔當中第 n 個詞語 (下標 $(m,n) = i$) 的條件概率，其中定義 $-i$ 表示去除 i 詞語，$\vec{w} = \{w_i = t, \vec{w}_{-i}\}$：

$$\begin{aligned}
p(z_i = k|\vec{z}_{-i}, \vec{w}) &\propto p(z_i = k, w_i = t|\vec{z}_{-i}, \vec{w}_{-i}) \\
&= \int p(z_i = k, w_i = t, \vec{\vartheta}_m, \vec{\varphi}_k|\vec{z}_{-i}, \vec{w}_{-i})\mathrm{d}\vec{\vartheta}_m \mathrm{d}\vec{\varphi}_k \\
&= \int p(z_i = k, \vec{\vartheta}_m|\vec{z}_{-i}, \vec{w}_{-i}) \cdot p(w_i = t, \vec{\varphi}_k|\vec{z}_{-i}, \vec{w}_{-i})\mathrm{d}\vec{\vartheta}_m \mathrm{d}\vec{\varphi}_k \\
&= \int p(z_i = k|\vec{\vartheta}_m)p(\vec{\vartheta}_m|\vec{z}_{-i}, \vec{w}_{-i}) \cdot p(w_i = t|\vec{\varphi}_k)p(\vec{\varphi}_k|\vec{z}_{-i}, \vec{w}_{-i})\mathrm{d}\vec{\vartheta}_m \mathrm{d}\vec{\varphi}_k \\
&= \int p(z_i = k|\vec{\vartheta}_m)Dir(\vec{\vartheta}_m|\vec{n}_{m,-i} + \vec{\alpha})\mathrm{d}\vec{\vartheta}_m \\
&\quad \cdot \int p(w_i = t|\vec{\varphi}_k)Dir(\vec{\varphi}_k|\vec{n}_{k,-i} + \vec{\beta})\mathrm{d}\vec{\varphi}_k \qquad -(*) \\
&= \int \vartheta_{m,k} Dir(\vec{\vartheta}_m|\vec{n}_{m,-i} + \vec{\alpha})\mathrm{d}\vec{\vartheta}_m \cdot \int \varphi_{k,t} Dir(\vec{\varphi}_k|\vec{n}_{k,-i} + \vec{\beta})\mathrm{d}\vec{\varphi}_k \\
&= E(\vartheta_{m,k}) \cdot E(\varphi_{k,t}) \\
&= \hat{\vartheta}_{m,k} \, \hat{\varphi}_{k,t}
\end{aligned}$$

- $(*)$Posterior of θ and ϕ：

其中 $\vec{\alpha}$ 是控制 $\vec{\theta}_m$ 的參數，與取得的 $z_{m,n}$ 獨立；$\vec{\beta}$ 是控制 $\vec{\varphi}_k$ 的參數，與取得的 $w_{m,n}$ 獨立。因此，對於參數 $\vec{\theta}_m, \vec{\varphi}_k$ 的後驗機率計算分別為：

$$
\begin{aligned}
p(\vec{\vartheta}_m|\vec{z}_m, \vec{\alpha}) &= \frac{p(\vec{\vartheta}_m, \vec{z}_m, \vec{\alpha})}{p(\vec{z}_m, \vec{\alpha})} \\
&= \frac{p(\vec{\alpha})p(\vec{\vartheta}_m, \vec{z}_m|\vec{\alpha})}{p(\vec{\alpha})p(\vec{z}_m|\vec{\alpha})} \\
&= \frac{p(\vec{z}_m|\vec{\vartheta}_m, \vec{\alpha})p(\vec{\vartheta}_m|\vec{\alpha})}{p(\vec{z}_m|\vec{\alpha})} \\
&= \frac{p(\vec{z}_m|\vec{\vartheta}_m)p(\vec{\vartheta}_m|\vec{\alpha})}{p(\vec{z}_m|\vec{\alpha})} \\
&= \frac{\prod_{n=1}^{N_m} p(z_{m,n}|\vec{\vartheta}_m)p(\vec{\vartheta}_m|\vec{\alpha})}{\int \prod_{n=1}^{N_m} p(z_{m,n}|\vec{\vartheta}_m)p(\vec{\vartheta}_m|\vec{\alpha}) \mathrm{d}\vec{\vartheta}_m} \\
&= \frac{\Delta(\vec{\alpha})}{\Delta(\vec{n}_m + \vec{\alpha})} \prod_{k=1}^{K} \vartheta_{m,k}^{n_m^{(k)}} \frac{1}{\Delta(\vec{\alpha})} \prod_{k=1}^{K} \vartheta_{m,k}^{\alpha_k - 1} \\
&= \frac{1}{\Delta(\vec{n}_m + \vec{\alpha})} \prod_{k=1}^{K} \vartheta_{m,k}^{n_m^{(k)} + \alpha_k - 1} \\
&= Dir(\vec{\vartheta}_m|\vec{n}_m + \vec{\alpha})
\end{aligned}
$$

$$
\begin{aligned}
p(\vec{\varphi}_k|\vec{z}, \vec{w}, \vec{\beta}) &= \frac{p(\vec{\varphi}_k, \vec{z}, \vec{w}, \vec{\beta})}{p(\vec{z}, \vec{w}, \vec{\beta})} \\
&= \frac{p(\vec{z}, \vec{\beta})p(\vec{\varphi}_k, \vec{w}|\vec{z}, \vec{\beta})}{p(\vec{z}, \vec{\beta})p(\vec{w}|\vec{z}, \vec{\beta})} \\
&= \frac{p(\vec{\varphi}_k, \vec{w}|\vec{z}, \vec{\beta})}{p(\vec{w}|\vec{z}, \vec{\beta})} \\
&= \frac{p(\vec{w}|\vec{\varphi}_k, \vec{z}, \vec{\beta})p(\vec{\varphi}_k|\vec{z}, \vec{\beta})}{p(\vec{w}|\vec{z}, \vec{\beta})} \\
&= \frac{p(\vec{w}|\vec{\varphi}_k, \vec{z})p(\vec{\varphi}_k|\vec{\beta})}{p(\vec{w}|\vec{z}, \vec{\beta})} \\
&= \frac{\prod_{\{i:z_i=k\}} p(\vec{w}|\vec{\varphi}_k)p(\vec{\varphi}_k|\vec{\beta})}{\int \prod_{\{i:z_i=k\}} p(\vec{w}|\vec{\varphi}_k)p(\vec{\varphi}_k|\vec{\beta}) \mathrm{d}\varphi_k} \\
&= \frac{\Delta(\vec{\beta})}{\Delta(\vec{n}_k + \vec{\beta})} \prod_{t=1}^{V} \varphi_{k,t}^{n_k^{(t)}} \frac{1}{\Delta(\vec{\beta})} \prod_{t=1}^{V} \varphi_{k,t}^{\beta_k - 1} \\
&= \frac{1}{\Delta(\vec{n}_k + \vec{\beta})} \prod_{t=1}^{V} \vartheta_{k,t}^{n_k^{(t)} + \beta_k - 1} \\
&= Dir(\vec{\varphi}_k|\vec{n}_k + \vec{\beta})
\end{aligned}
$$

經上述計算，可得 $\vec{\theta}_m, \vec{\phi}_k$ 的後驗機率分布也服從 Dirichlet 分布：

$$p(\vec{\vartheta}_m|\vec{z}_m,\vec{\alpha}) = Dir(\vec{\vartheta}_m|\vec{n}_m+\vec{\alpha})$$
$$p(\vec{\varphi}_k|\vec{z},\vec{w},\vec{\beta}) = Dir(\vec{\varphi}_k|\vec{n}_k+\vec{\beta})$$

- Exepectation of θ and ϕ：

Dirichlet distribution expection: $E[Dir(\alpha)] = \frac{\alpha_i}{\sum_i \alpha_i}$，所以得其 θ and ϕ 的期望為：

$$\vartheta_{m,k} = \frac{n_m^{(k)}+\alpha_k}{\sum_k (n_m^{(k)}+\alpha_k)}$$
$$\varphi_{k,t} = \frac{n_k^{(t)}+\beta_t}{\sum_t (n_k^{(t)}+\beta_t)}$$

最後將 $\vartheta_{m,k}, \varphi_{k,t}$ 代入 $p(z_i=k|\vec{z}_{-i},\vec{w}) \propto p(z_i=k, w_i=t|\vec{z}_{-i},\vec{w}_{-i})$，可得：

$$p(z_i=k|\vec{z}_{-i},\vec{w}) \propto \frac{n_{m,-i}^{(k)}+\alpha_k}{\sum_{k=1}^{K}(n_{m,-i}^{(k)}+\alpha_k)} \cdot \frac{n_{k,-i}^{(t)}+\beta_t}{\sum_{t=1}^{V}(n_{k,-i}^{(t)}+\beta_t)}$$

其等號右方是為 $p(topic|document)\cdot p(word|document)$，其路徑可由下方表示：

$$doc \xrightarrow{\vartheta_{m,k}} topic \xrightarrow{\varphi_{k,t}} word$$

- Gibbs Sampling:

$$doc \xrightarrow{\vartheta_{m,k}} topic \xrightarrow{\varphi_{k,t}} word$$

Gibbs Sampling 將會在初始設定好的 topic 數量 k 上對上圖路徑 (k 條) 進行採樣，至 $\vec{z}$ 實現收斂，而每篇文檔對應的主題分布 $\vec{\vartheta}_m$ 及每個主題下對應的詞語分布 $\vec{\varphi}_k$ 也就達到收斂。

4 Data Analyis

此處以 $tf-idf$ 的值最高的 30 個詞語進行以下分析 (除去 Latent Dirichlet Allocation)。

4.1 K-Means

- Elbow Method

1. 計算在不同個 K 下的不同分群函式
2. 對每個 K，我們計算群組內的組內距離平方和 (Within-cluster Sum of Square)
3. 以 K 為橫軸，WSS 為縱軸作圖
4. 找出上圖中使坡度變化最大的 K 值

- Average Silhouette Method(平均側影法)

此演算法根據每個資料點 (i) 的內聚力和分散力去衡量分群的效果，首先側影係數 (Silhouette Coefficient) 如下：

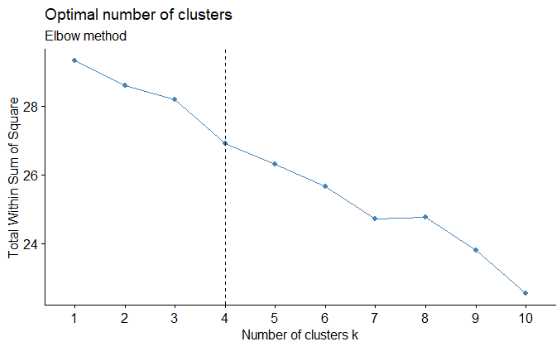
$$s(i) = \frac{b(i) - a(i)}{max\{a(i),\ b(i)\}}$$

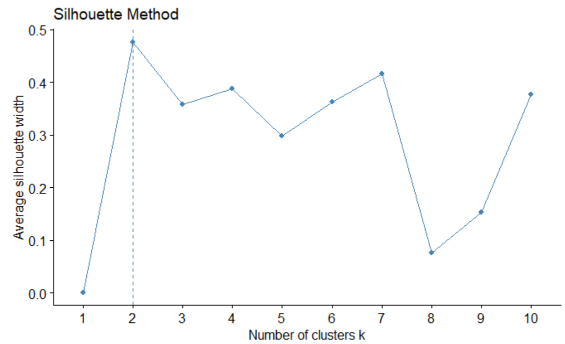
$a(i)$：資料點 i 與群內其他資料點的平均距離

$b(i)$：資料點 i 與其他群內資料點的平均距離，取最小值

$s(i)$：側影係數，可視為資料點 i 在它所屬的群內是否適當的指標

1. 計算在不同個 K 下的不同分群函式
2. 對每個 K，我們計算出 K 個各集群內的側影係數平均
3. 將以上 K 個不同集群中的平均側影值再做平均，得全部資料的平均側影寬度 (average silhouette width)
4. 找出使平均側影寬度最大化的 K 值





經過 Elbow Method(左圖) 計算所得結果並不彰顯最有效率的 K 值，接著利用平均側影法 (Average Silhouette Method) 尋找最佳 K 值；如右圖所示，$K = 2$ 時具最大的平均側影寬度，代入 K-means 的分類器後，得到以下的結果：

	Actual		
Predicted	1	2	Row Total
1	84 0.494 0.026	86 0.506 0.020	170 0.022
2	3187 0.428 0.974	4256 0.572 0.980	7443 0.978
Column Total	3271 0.430	4342 0.570	7613

其中分群準確度為 0.5701，但從上圖 Predicted=2 的 Row Total 可發現，幾乎所有資料皆被預測為第二分群，模型配適不佳。

4.2 Gaussian Mixture Model

首先，設置 2~9 個混合分布、運用 14 種不同模型的情況下，根據 BIC（Bayesian Information Criteria）繪製的曲線，並以最大化 BIC 值作為分群數目的評斷標準，得出下列 BIC Plot。

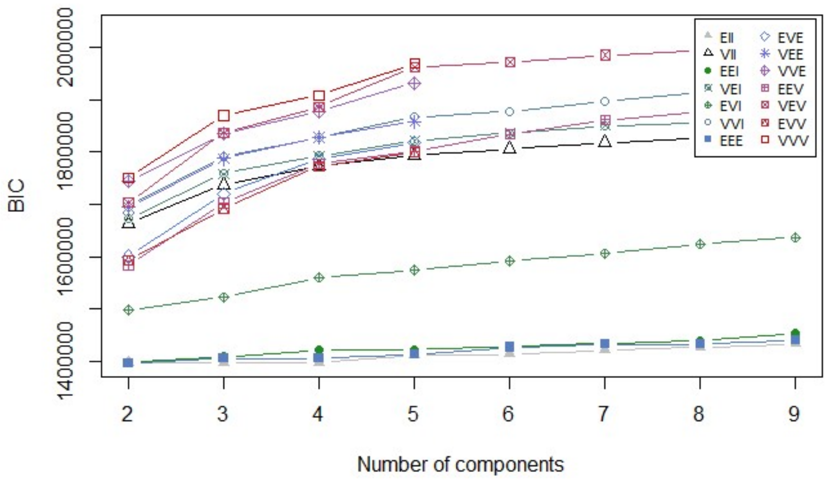


Figure 8: BIC Plot of GMM

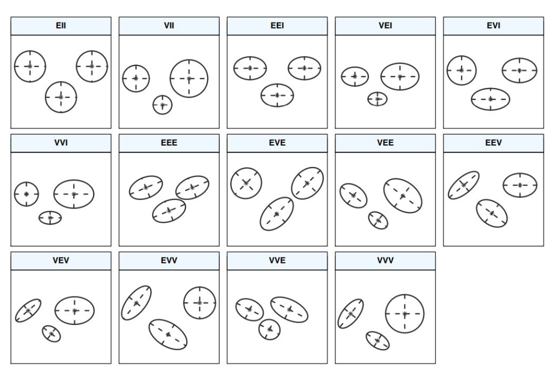
在 mclust 中，其 BIC 的定義如下：

$$BIC \equiv 2\ loglik_M(x,\theta) - (\#params)_M log(n)$$

與常定義的差一個負號，故此處要選擇 BIC 較大的模型。

以下為 14 種預設模型的細節，且為了方便理解每個欄位代表的涵義，右圖為這些模型假設在分成三群的前提下二維空間內的形狀：

Model	Σ_k	Distribution	Volume	Shape	Orientation
EII	$\lambda \boldsymbol{I}$	Spherical	Equal	Equal	—
VII	$\lambda_k \boldsymbol{I}$	Spherical	Variable	Equal	—
EEI	$\lambda \boldsymbol{A}$	Diagonal	Equal	Equal	Coordinate axes
VEI	$\lambda_k \boldsymbol{A}$	Diagonal	Variable	Equal	Coordinate axes
EVI	$\lambda \boldsymbol{A}_k$	Diagonal	Equal	Variable	Coordinate axes
VVI	$\lambda_k \boldsymbol{A}_k$	Diagonal	Variable	Variable	Coordinate axes
EEE	$\lambda \boldsymbol{DAD}^\top$	Ellipsoidal	Equal	Equal	Equal
EVE	$\lambda \boldsymbol{DA}_k\boldsymbol{D}^\top$	Ellipsoidal	Equal	Variable	Equal
VEE	$\lambda_k \boldsymbol{DAD}^\top$	Ellipsoidal	Variable	Equal	Equal
VVE	$\lambda_k \boldsymbol{DA}_k\boldsymbol{D}^\top$	Ellipsoidal	Variable	Variable	Equal
EEV	$\lambda \boldsymbol{D}_k\boldsymbol{AD}_k^\top$	Ellipsoidal	Equal	Equal	Variable
VEV	$\lambda_k \boldsymbol{D}_k\boldsymbol{AD}_k^\top$	Ellipsoidal	Variable	Equal	Variable
EVV	$\lambda \boldsymbol{D}_k\boldsymbol{A}_k\boldsymbol{D}_k^\top$	Ellipsoidal	Equal	Variable	Variable
VVV	$\lambda_k \boldsymbol{D}_k\boldsymbol{A}_k\boldsymbol{D}_k^\top$	Ellipsoidal	Variable	Variable	Variable



Σ_k：第 K 個群組共變異矩陣的計算方式。

Distribution：Spherical 為球體、Diagonal 為對角圖形、Ellipsoidal 為橢圓體

Volume：圖形的大小，視不同分群的大小是否均一

Shape：視不同分群的形狀是否均一

Orientation：視不同分群內導向是否均一，抑或是與 p 維度的座標軸一致

在本次分析將設定 $G = 2$，將 G 代入 GMM 分類器，得到以下的結果：

Predicted	Actual 1	2	Row Total
1	730 0.674 0.223	353 0.326 0.081	1083 0.142
2	2541 0.389 0.777	3989 0.611 0.919	6530 0.858
Column Total	3271 0.430	4342 0.570	7613

其中分群準確度為 0.6199，較 K-means 的分群準確度高約 4%；但從上圖 Actual=1 的欄位中可發現，此群資料的預測將許多分類為 1 的錯誤歸類為 2，在預測第一分群表現並非理想。

接著將眾多變數拆成兩張圖個別解釋「該變數中 2 個分組之間的差異程度」：

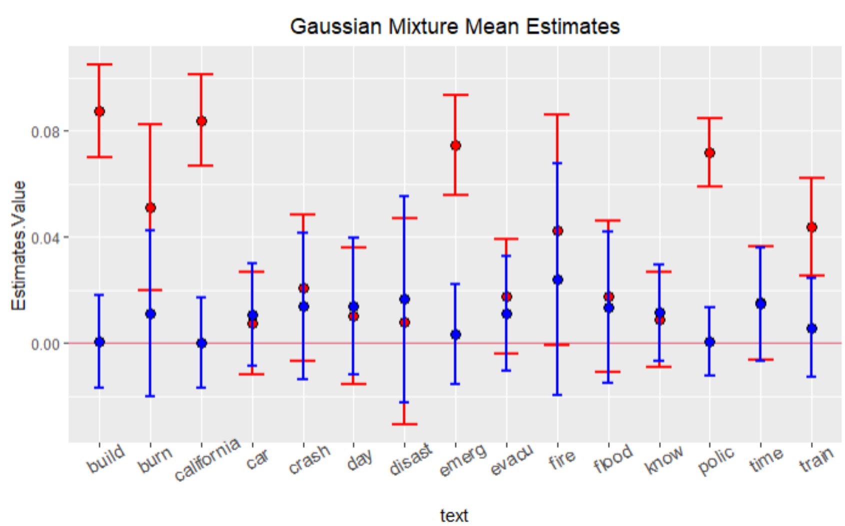


Figure 9: Mean Estimates with 95% Confidence Interval(First 15)

在上圖中，單詞「**buildings**」、「**california**」、「**emergency**」、「**police**」、「train」的分群效果明顯，而大部分變數的信賴區間都有重疊的現象。

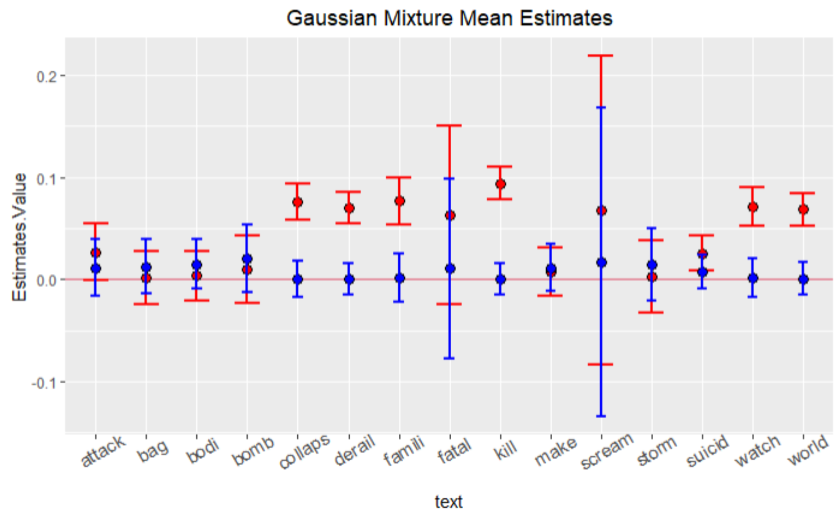


Figure 10: Mean Estimates with 95% Confidence Interval(Next 15)

在上圖中，值得注意的是單詞「**collapse**」、「**derail**」、「**kill**」的分群效果較明顯，其中以「**scream**」的信賴區間全距最長，故推測該變數中資料點變異度很大。

4.3 Logistic

在未選定 Cutpoint(預設為 0.5) 的狀況下建立模型取得的準確率為 0.7196671。而後以 0.01 為單位在 0.1 與 0.9 之間尋找 Cutpoint 找出其值為 0.58，並將測試集資料帶入模型進行預測可得到以下結果，其精確度為 0.7232。

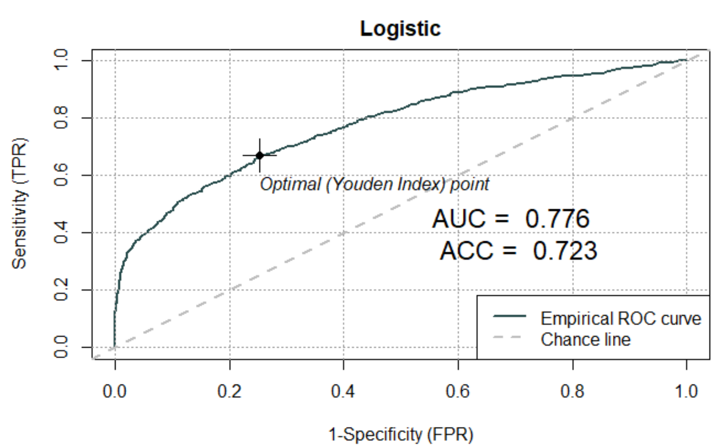


Figure 11: ROC Curve of Logistic

4.4 Random Forest

首先以未調整任何參數的情況下建構模型，並藉此判斷可達到最低誤差的最佳 tree size 大約落在 251；而後分別對 3 個參數：mtry, node size, sample size 做 grid search，並以 Out-of-Bag Error 作為標準進行參數調控。

下表為做完 grid search 候選取出來，根據 OOB error rate，選出表現最好的參數組合。

模型參數	參數值
ntree	251
mtry	5
Node_size	17
Sample_size	0.7

配適後得出以下結果，其精確度為 0.7643。

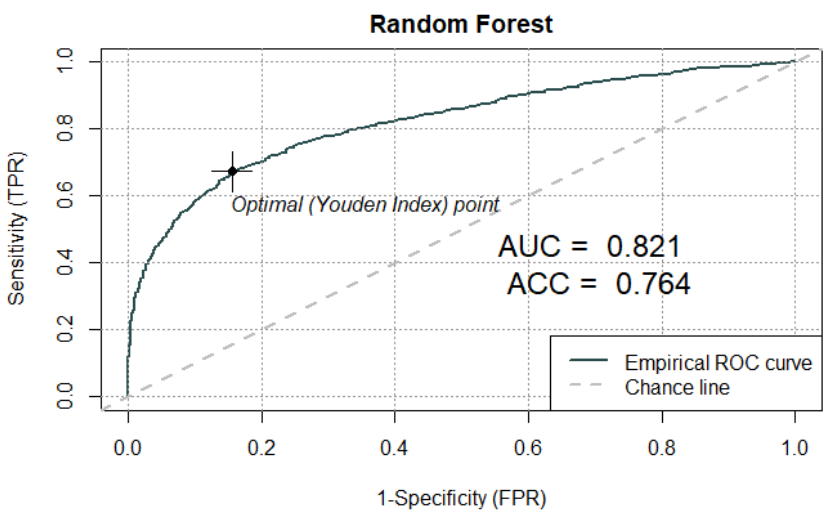


Figure 12: ROC Curve of Random Forest

4.5 Latent Dirichlet Allocation

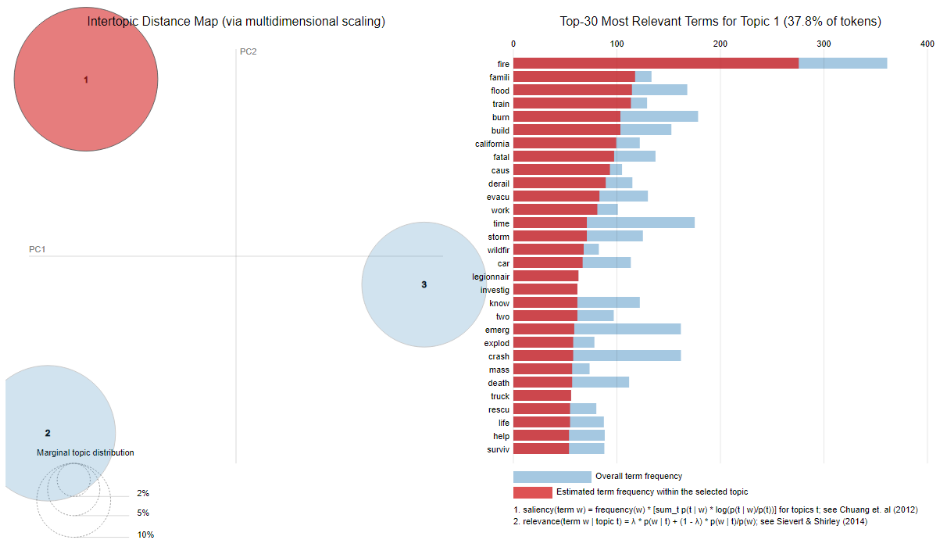


Figure 13: Topic Cluster 1.

在主題一中可以發現到，*fire*, *flood*, *burn*, *fatal*, *evacu(ation)*, *storm*, *wildfir(e)* 等詞語在此主題出現的頻率比例較高，此主題應與災害相關。

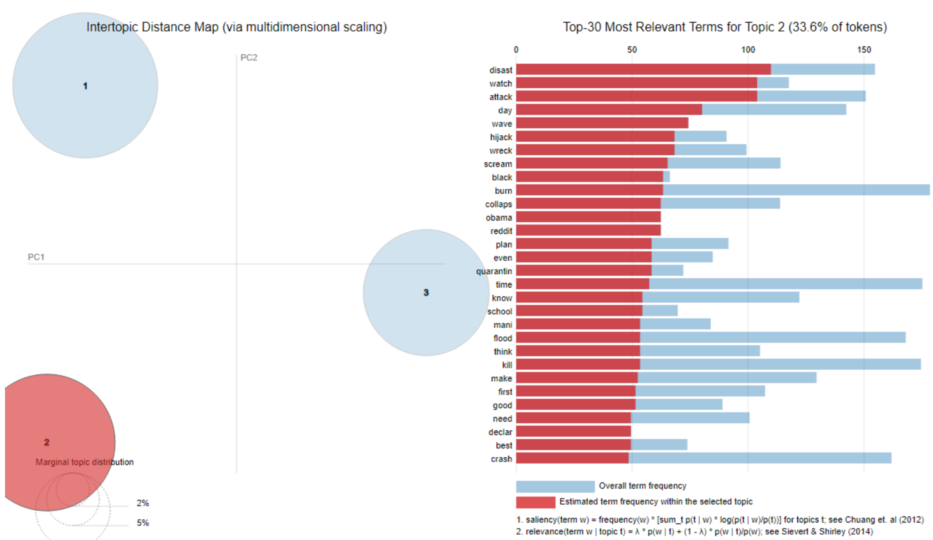


Figure 14: Topic Cluster 2.

在主題二中，*disaster*, *attack*, *hijack*, *wreck*, *burn*, *collaps(e)*, *Obama*, *reddit* 等詞語在此主題出現的頻率比例較高，此主題類似於主題一與災害相關，但又有關於政治人物、新聞媒體的字眼，因此此主題應與新聞播報有關。

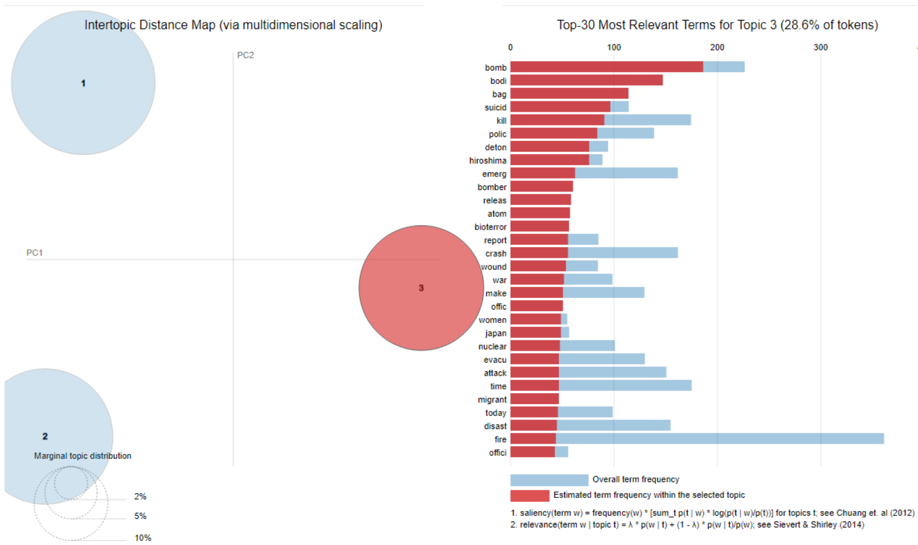


Figure 15: Topic Cluster 3.

在主題三中，$bomb$, $body$, $suicid(e)$, $kill$, $polic(e)$, $emerg(ency)$, $bioterror$ 等詞語在此主題出現的頻率比例較高，此主題應與社會秩序有關，並且「警察」、「炸彈」、「殺害」等字眼的頻繁出現，所以此主題應偏向刑事案件、恐怖攻擊。

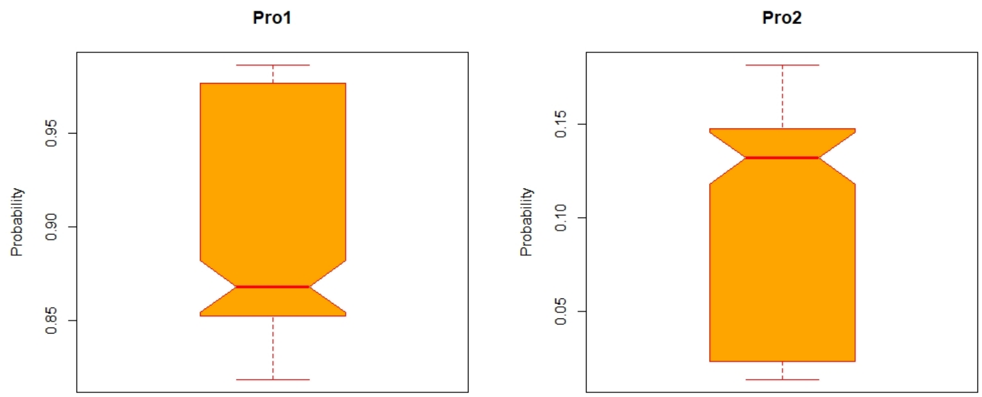
5 Simulation Study

5.1 Bootstrap Accuracy

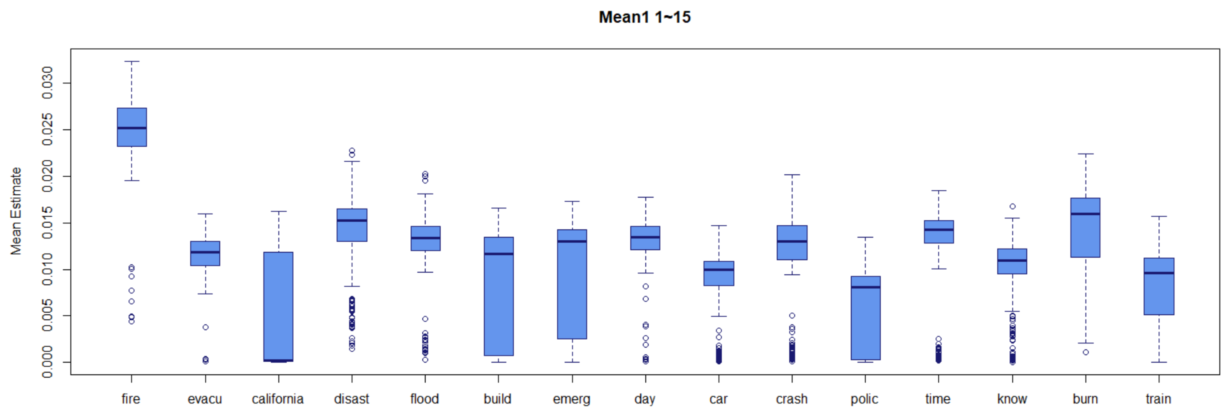
對資料進行 200 重抽樣的模擬，由分類結果的分佈可以明顯看出 GMM 的準確度明顯優於 k-means，而若個別來看，k—means 的結果分佈更爲集中，多數結果落在 0.56-0.58 之間，相對的 GMM 的結果浮動較大，多數結果落在 0.58-0.62 之間。若加入 Logistic Regression 進行比較會發現，Logistic Regression 的結果顯著的優於其餘而這，其原因在於 Logistic Regression 屬於 supervised learning 有加入 label 的額外資料，準確率自然較高。而 Random Forest 因模型建構時便已使用 bootstrap 進行資料選取了，故沒有對其進行重新抽樣的模擬，在此章亦不多加討論。

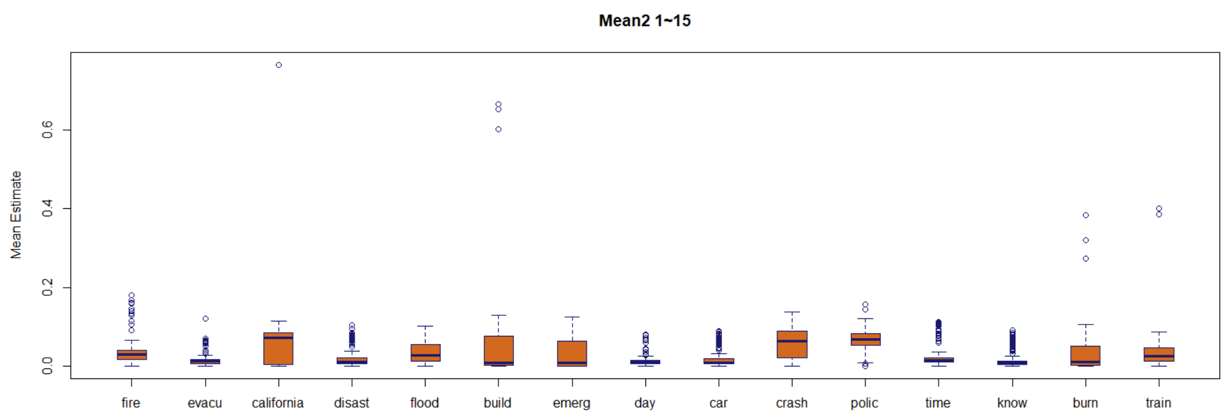
5.2 Bootstrap Parameters

由 $\hat{\pi}$ 參數可以明顯看出兩個群組的幾率差異及其顯著，群組 1 的幾率高達 $0.85 \sim 0.95$，而群組而只有 $0.5 \sim 0.15$，但原始資料中，兩個類別的數量是相近的，這便導致了 GMM 模型的準確率並不理想。推測原因可能在於資料原始資料並不符合常態分配導致了誤差。

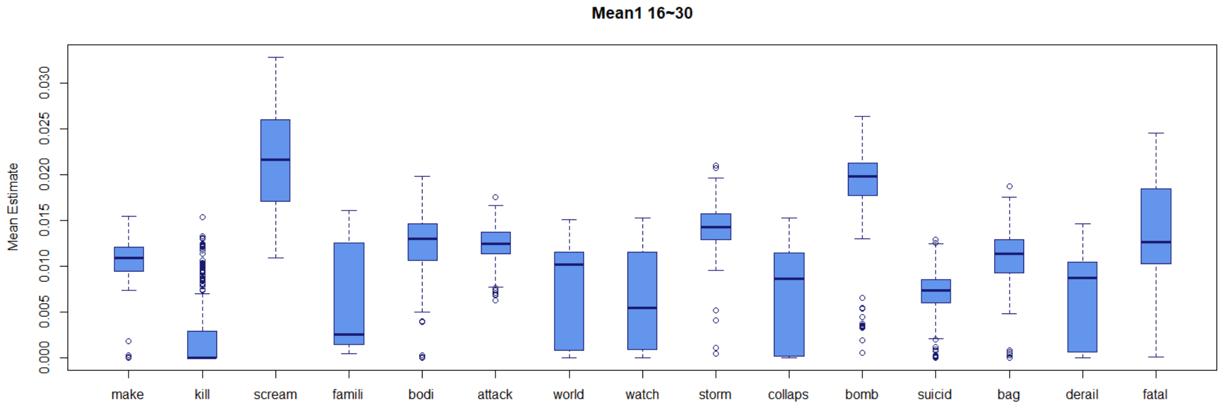


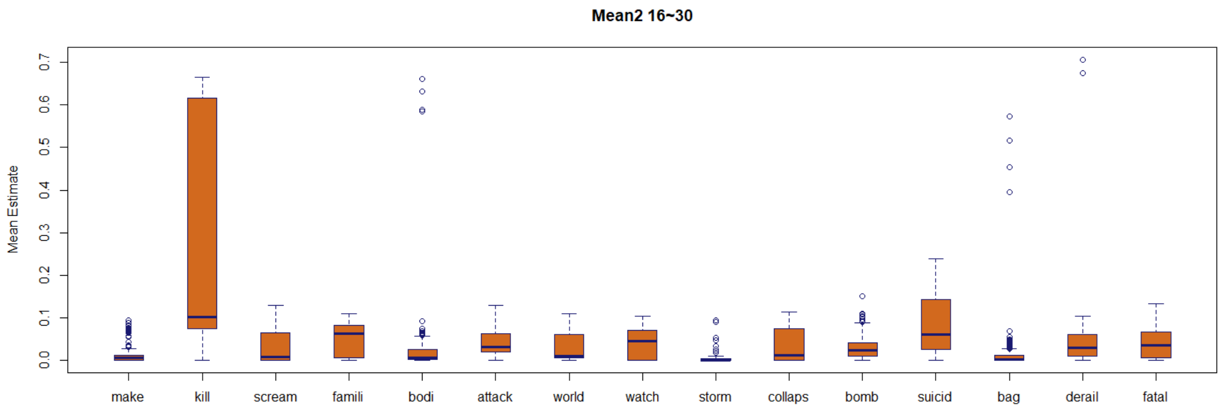
比較前 15 個詞，由第一個群組的各個平均值可以看出多數詞匯的 TF-IDF 值坐落在 0-0.015 附近，表示特色并不明顯，其中「**fire**」雖然只有 0.025 左右的數值，卻也是群組中明顯較高的。而在第二群組中，大多數值落在 0~0.1 附近，其數值略大與第一群組，分佈情形則是明顯比第一群廣，此外少數如「**bulid**」，「**California**」有離群值可以到達 0.6 高點，「**burn**」，「**train**」也有 0.4 的離群值高點。



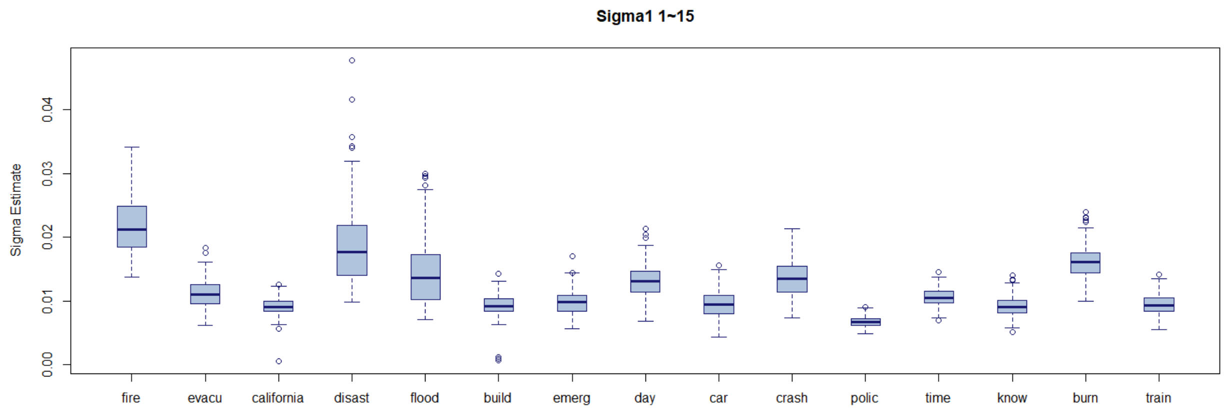


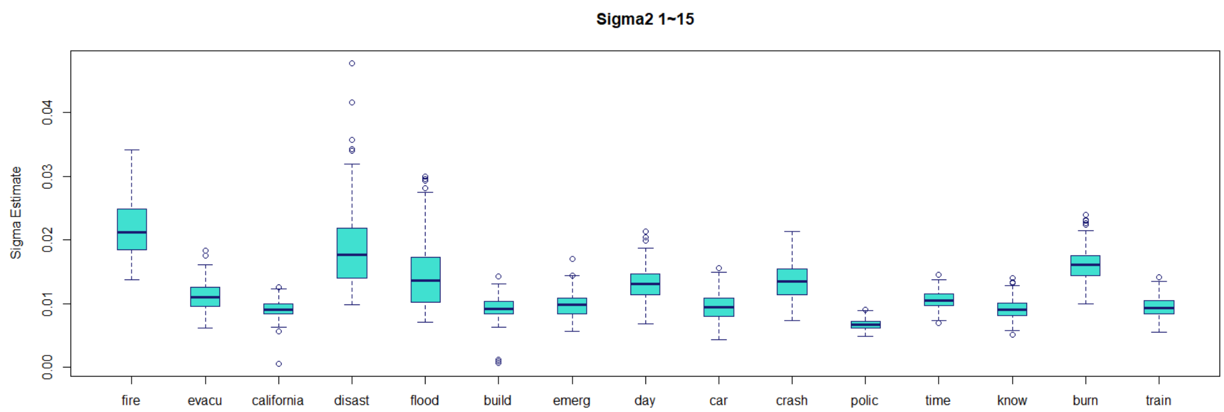
剩下的 15 個詞匯中，第一個群集裡「**scream**」，「**bomb**」，有較高的數值，達到 0.02，而第二群集中，分佈同樣較第一群廣，其中「**kill**」，「**bodi**」，「**bag**」，「**derail**」甚至有到達 0.6~0.7 的離群值，又以「**kill**」離散的程度最强烈，且與第一群有明顯差異。



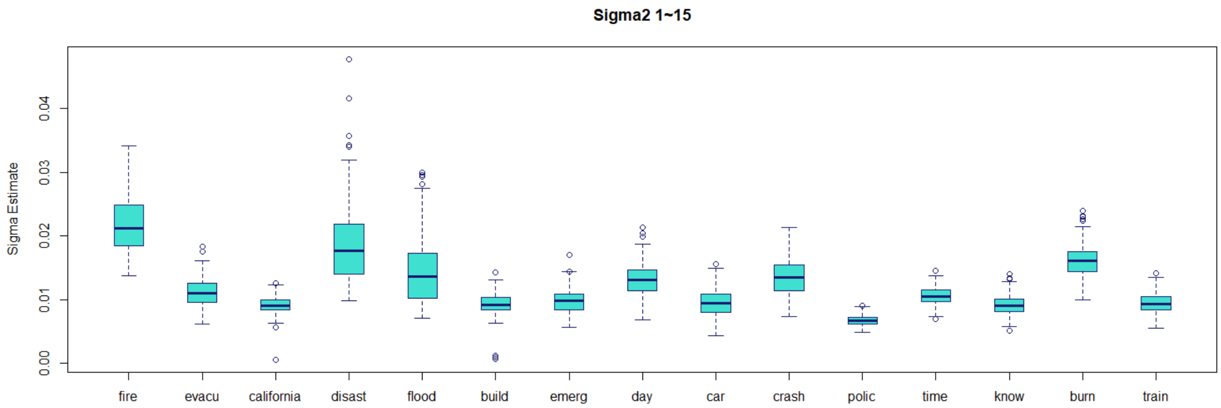


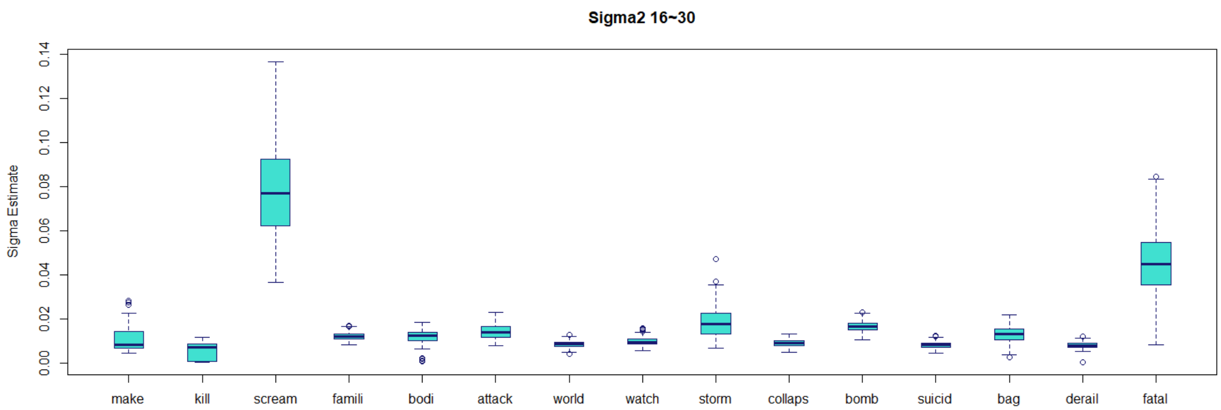
由於共變異數矩陣中，所有詞匯的共變異數數值皆明顯較小，落在 $10^{-4\sim10}$-8，與變異數差距明顯，故一下僅對其各自的變異數進行討論。第一群組中變異數大多數落在 0.01 附近，可見浮動程度極小，而「**fire**」「**disast**」「**flood**」有較高的數值與較分散的分佈情況，其中「**disast**」有達到 0.04 左右的離散值出現。





在後 15 個詞匯中變異數也多數落在 0.01 附近，「**scream**」「**storm**」有非常顯著的突出，無論是數值還是分散狀況都明顯較高。此時也可以看出，突出的數值多為災害相關字眼「**fire**」「**flood**」「**storm**」等，這些偶發事件使得其出現頻率的變動較大。





6 Conclusion

根據以上的分析，儘管隨機森林中利用了 Bootstrap 進行重抽樣，在預測精準度上會較高，但無論是非監督式學習的高斯混合模型、K 平均演算法，或是監督式學習的羅吉斯回歸與隨機森林，四者的表現都不如預期，推斷其可能原因為：倘若應用 tfidf—將文字轉換成變數使用—進行文字探勘會因為某些文檔沒有某字詞而使其絕大部分數值為零，導致訊息不足夠；而其中監督式學習演算法表現相對較好應歸因於：資料集匯入時包含 labeled data，多了一項資訊可以參照，故在準確率上會較非監督式高。

在統計計算中，模型需要對資料的分配達到一定的配適，或是資料需要對模型的假設 (分配) 達到配適，此模型才可以從資料信息找到規律或統計量，並進行預測。對於文字資料，Latent Dirichlet Allocation 配適程度較上述模型好，其一因為複雜度高、其二因為其與文字生成的形式較為配適；但因 Latent Dirichlet Allocation 並不適合預測資料的真假，其在分析文檔的主題分布、詞語分布會更為恰當。

7 Appendix

7.1 工作分配

組員	工作分配
蘇柏庄	EDA、K-Means、GMM、Latent Dircihlet、書面報告、PPT
吳岱錡	Logistic、Random Forest、Data Analysis、書面報告
陳威宇	K-Means、GMM、Simulation、PPT
王彥翔	tf-idf
蔡雅欣	EDA

7.2 文獻參考

- Ramos, J. (2003). Using tf-idf to determine word relevance in document queries. Proceedings of the First Instructional Conference on Machine Learning, 242, 133–142. Piscataway, NJ.

- Trstenjak, B., Mikac, S., & Donko, D. (2014). KNN with TF-IDF based framework for text categorization. Procedia Engineering, 69, 1356–1364.

- Blei, David M.; Ng, Andrew Y.; Jordan, Michael I. (2003). Latent Dirichlet allocation. Journal of Machine Learning Research. 3 (4–5): pp. 993–1022.