

統計應用與計算基礎實作手冊

韓明澄

January 11, 2024



目錄

序

vi

1 L^AT_EX 基本工具手冊	1
1.1 基本模板	1
1.1.1 標題	1
1.1.2 文章內容的排版	2
1.1.3 圖片表格標題	4
1.1.4 註腳與程式碼樣子	5
1.2 文字	6
1.2.1 字體樣式與顏色	6
1.2.2 字體大小	6
1.2.3 字顏色	7
1.2.4 結合使用	7
1.3 數學式子	8
1.3.1 基本運算符號	8
1.3.2 數學符號	8
1.4 圖片	11
1.4.1 加入圖片	11
1.4.2 多張圖片並排	13
1.4.3 文字旁的子圖	16

1.5 表格	17
1.5.1 基礎表格	17
1.5.2 特別表格	21
1.6 結論	27
2 Python 的數學製圖	29
2.1 Matplotlib 函數圖形的繪製	29
2.1.1 特殊三角函數與極限	29
2.1.2 函式在不同 α 值下的表現	31
2.1.3 函式震盪幅度	33
2.1.4 函數值無定義	35
2.1.5 函數、反函數與對稱線	36
2.1.6 常態分配	37
2.1.7 函數找根值	38
2.1.8 函數的右漸近線與最大值	39
2.1.9 階梯函數	41
2.1.10 單位圓	42
2.1.11 正方形	44
2.2 專題：比較審斂法	46
2.3 結論	49
3 Python 分配圖型多樣性	51
3.1 不同連續與離散分配中的多種可能性	51
3.1.1 常態分配	51
3.1.2 t 分配	52
3.1.3 F 分配	53
3.1.4 柯西分配	54
3.1.5 卡方分配	55

3.1.6 貝塔分配	56
3.1.7 Gamma 分配	57
3.1.8 二項分配	58
3.1.9 柏松分配	59
3.2 亂數分配分析	59
3.2.1 亂數生成韋伯分配	59
3.2.2 亂數生成柏松分配	60
3.3 抽樣分配	61
3.3.1 中央極限定理	62
3.3.2 卡方分配與 F 分配關係	63
3.3.3 Gamma 分配與 Beta 分配	63
3.3.4 二項分配累積性	64
3.4 四數字隨機抽樣	65
3.5 結論	66
4 監督式機器學習迴歸模型	67
4.1 機器學習工具	67
4.2 模型介紹	68
4.2.1 簡單線性迴歸	68
4.2.2 增廣迴歸	69
4.2.3 邏輯斯迴歸	69
4.3 回歸模型特性與優劣	70
4.3.1 匯入資料與簡單說明	70
4.3.2 兩群組之距離近	71
4.3.3 兩群組之距離遠	72
4.3.4 共變數矩陣較大	73
4.3.5 共變異數矩陣較小	74

4.3.6 樣本數	75
4.3.7 樣本相依	76
4.3.8 比較	77
4.3.9 兩類資料邏輯斯迴歸模型	78
4.3.10 三類資料邏輯斯迴歸模型	79
4.4 訓練資料與測試	80
4.5 結論	80
5 監督式學習器與應用	81
5.1 線性判別分析 (LDA)	81
5.2 二次判別分析 (QDA)	84
5.3 K 鄰近模型 (KNN)	85
5.4 兩群資料之學習器評比	87
5.4.1 兩群共變數矩陣相同	87
5.4.2 兩群資料共變數不同且正相關	88
5.4.3 兩群資料共變數不同且相關性一正一負	89
5.4.4 兩群資料共變數不同且負相關	91
5.4.5 兩群平均數接近且變異數不同	93
5.4.6 兩群資料小結	94
5.5 三群資料學習器評比	96
5.5.1 三群共變數矩陣相同	96
5.5.2 三群兩組共變數相同且正相關	97
5.5.3 三群兩組共變數相同且負相關	98
5.5.4 三群共變數皆不同	100
5.5.5 三群資料小結	101
5.6 結論	102
6 監督式類神經網路學習器	103

6.1	類神經網路介紹	103
6.1.1	類神經網路基本概念	103
6.1.2	前饋式類神經網路	104
6.2	機械手臂簡介與實測	106
6.2.1	逆運動方程式	106
6.2.2	機械手臂實作	107
6.2.3	小節	109
6.3	資料生成與預測	110
6.3.1	訓練與測試資料生成	110
6.3.2	神經元個數與樣本個數	110
6.3.3	圖像辨識	112
6.4	結論	115
7	蒙地卡羅模擬實驗：學習器的評比	117
7.1	兩群資料綜合學習器評比	117
7.1.1	兩群資料共變數矩陣皆相同	117
7.1.2	兩群資料共變數矩陣不同且兩者正相關	119
7.1.3	兩群資料共變數矩陣不同且矩陣關係一正一負	120
7.1.4	兩群資料共變數矩陣不同且兩者負相關	122
7.1.5	兩群資料平均數接近且共變數矩陣不同	123
7.2	三群資料綜合學習器評比	125
7.2.1	三群資料共變數矩陣皆相同	125
7.2.2	三群資料共變數矩陣兩組相同且正相關	126
7.2.3	三群資料共變數矩陣兩組相同且負相關	127
7.2.4	三群資料共變數矩陣都不同	129
7.3	結論	131

圖目錄

圖 1.1 標題	2
圖 1.2 註腳與程式碼樣子	5
圖 1.3 匯入 pdf 檔圖	12
圖 1.4 匯入 PNG 檔圖	12
圖 1.5 調整大小	12
圖 1.6 旋轉角度	13
圖 1.7 兩圖並排	14
圖 1.8 三圖並排	14
圖 1.9 四宮格排圖	15
圖 1.10 excel 做 latex 表	25
圖 2.1 $f(x) = \frac{\sin(x)}{x}, g(x) = \frac{\sin(x^2)}{x}$ 與 $h(x) = \frac{\sin^2(2x)}{x^2}$	30
圖 2.2 $\frac{e^{\alpha x}}{e^{\alpha x} + 1}$ 與 α 關係圖	31
圖 2.3 $\frac{e^{\alpha x}}{e^{\alpha x} + 1}$ 與其漸近線	33
圖 2.4 $f(x) = e^{\frac{-x}{10}} \sin(x)$ 震盪幅度	34
圖 2.5 $f(x) = \frac{1}{x-2}$	35
圖 2.6 $f(x) = \frac{1}{x-2}$ 漸近線	36
圖 2.7 $f(x) = x^3 + 2$ 與其反函數	37
圖 2.8 $f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-3)^2}{2}}$ 與 68–95–99.7 法則	38
圖 2.9 $f(x) = 3x^3 - x^4$ 的實數根與最大值	39

圖 2.10	$f(x) = 3x^3 - x^4$ 的實數根與最大值	39
圖 2.11	$f(x) = \frac{\ln x}{x^2}$ 右漸近線與最大值	40
圖 2.12	$f(x) = \begin{cases} 1, & 1 \leq x < 32, \\ 3 \leq x < 53, & 5 \leq x < 7 \end{cases}$	41
圖 2.13	變數法製作單位圓	42
圖 2.14	matplotlib 套件單位圓	44
圖 2.15	四條線法畫正方形	44
圖 2.16	定位四角畫正方形	45
圖 2.17	matplotlib 套件製作正方形	46
圖 2.18	$S_n = \sum_{k=1}^n \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}$	46
圖 2.19	$\gamma_n = S_n - \ln(n+1)$	47
圖 2.20	$\frac{1}{2}(1 - \frac{1}{n+1}) < \gamma_n < 1$	48
圖 3.1	常態分配 μ 與 σ 關聯	52
圖 3.2	t 分配與其自由度的關聯	53
圖 3.3	f 分配自由度	54
圖 3.4	柯西分配與常態	55
圖 3.5	多自由度卡方分配圖	56
圖 3.6	不同參數下的 Beta 分配	57
圖 3.7	Gamma 分配與參數關係	58
圖 3.8	二項分配 PMF 與 CDF	58
圖 3.9	poissonPMF 與 CDF	59
圖 3.10	亂數生成韋伯分配	60
圖 3.11	亂數柏松	61
圖 3.12	離散分配與中央極限定理	62
圖 3.13	卡方分配與 f 分配	63
圖 3.14	Gamma 與 Beta 近似關係	64

圖 3.15 二項分配累加性	65
圖 3.16 四數字隨機抽樣最大值	65
圖 3.17 四數字隨機抽樣	66
圖 4.1 匯入資料做迴歸	70
圖 4.2 群組平均差距小	72
圖 4.3 群組平均差距大	73
圖 4.4 群組變異數矩陣值大	74
圖 4.5 群組變異數矩陣值小	75
圖 4.6 樣本數 100vs200,500,1000 比較圖	76
圖 4.7 資料共變數矩陣相依	77
圖 4.8 兩組資料邏輯斯迴歸模型比較圖	78
圖 4.9 三組資料邏輯斯	79
圖 4.10 訓練與測試	80
圖 5.1 資料 la_1 LDA 分界線	84
圖 5.2 資料 la_1 QDA 分界線	85
圖 5.3 資料 la_1 ,K=15 時 KNN 分析	87
圖 5.4 共變數矩陣相同下 LDA,QDA,KNN=15 與 KNN=20 比較圖	89
圖 5.5 共變數矩陣不同正相關下 LDA,QDA,KNN=15 與 KNN=20 比較圖	90
圖 5.6 共變數矩陣不同且相關性一正一負下各學習器比較圖	91
圖 5.7 共變數矩陣不同且負相關下各學習器比較圖	92
圖 5.8 共變數矩陣不同且正相關下個學習器比較圖	94
圖 5.9 不同種學習器：三群共變數矩陣相同	97
圖 5.10 不同種學習器：三群共變數矩陣兩個相同且正相關	98
圖 5.11 不同種學習器：三群共變數矩陣兩個相同且負相關	99

圖 5.12 不同種學習器：三群共變數矩陣兩個相同且負相關	101
圖 6.1 神經網路架構示意圖	104
圖 6.2 前饋式類神經網路架構圖	105
圖 6.3 逆運動方程式座標平面圖	106
圖 6.4 機械手臂位置圖範例	107
圖 6.5 不同神經元數目下的 Rmse 比較圖	108
圖 6.6 神經元數目 vs Rmse	109
圖 6.7 神經元 10 個下 lbfgs,sgd 與 adam 比較圖	109
圖 6.8 訓練與測試資料生成	110
圖 6.9 資料樣本數目 (100,500,1000) vs 神經元數目 (5,12,15)	111
圖 6.10 圖 6.9 對應訓練誤差趨勢圖	112
圖 6.11 數字 20*30 蒙太奇圖陣	113
圖 6.12 10 隱藏層混淆矩陣與測試誤差圖	113
圖 6.13 30 隱藏層混淆矩陣與測試誤差圖	114
圖 6.14 英文字 20*30 蒙太奇圖陣	114
圖 6.15 30 隱藏層英文混淆矩陣與測試誤差圖	115
圖 6.16 50 隱藏層英文混淆矩陣與測試誤差圖	115
圖 7.1 共變數矩陣相同下邏輯斯回歸模型、LDA、QDA、 KNN=5、KNN=15、ANN=10、ANN=30 比較圖	118
圖 7.2 共變數矩陣不同且正相關下邏輯斯回歸模型、LDA、 QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖	120
圖 7.3 共變數矩陣不同且關係一正一負下邏輯斯回歸模型、 LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比 較圖	121
圖 7.4 共變數矩陣不同且負相關下邏輯斯回歸模型、LDA、 QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖	123

圖 7.5 共變數矩陣不同且平均數相近下邏輯斯回歸模型、 LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比 較圖	124
圖 7.6 三群資料共變數矩陣相同下邏輯斯回歸模型、LDA、 QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖 . .	126
圖 7.7 三群資料共變數矩陣兩組相同且正相關下邏輯斯回歸 模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖	128
圖 7.8 三群資料共變數矩陣兩組相同且負相關下邏輯斯回歸 模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖	129
圖 7.9 三群資料共變數矩陣皆不同下邏輯斯回歸模型、 LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比 較圖	131

表目錄

表 1.1	2×2 表格	17
表 1.2	有瑕疵的基本的表格	18
表 1.3	基本的表格	18
表 1.4	旋轉表格	19
表 1.5	全表格填色	19
表 1.6	表格指定區域填色	20
表 1.7	表格行填色	20
表 1.8	表格列填色	21
表 1.9	兩個表格並列擺放	22
表 1.10	booktabs 套件改變表格格線粗細	23
表 1.11	booktabs 套件改變垂直間距	23
表 1.12	booktabs 斷線問題	24
表 1.13	長型表格	25
表 3.1	常態、t 和柯西分配比較表	55
表 3.2	Beta 分配比較表	57
表 5.1	相同共變數矩陣在不同學習器下的誤判率	88
表 5.2	不同共變數矩陣且正相關在不同學習器下的誤判率	88
表 5.3	不同共變數矩陣相關性一正一負下的誤判率	90
表 5.4	不同共變數矩陣且負相關在不同學習器下的誤判率	92

表 5.5 不同共變數矩陣且平均數接近在不同學習器下的誤判率	93
表 5.6 兩群預測資料誤判率比較	95
表 5.7 兩群訓練資料誤判率比較	95
表 5.8 三群：相同共變數矩陣在不同學習器下的誤判率	97
表 5.9 三群：兩組相同共變數矩陣且正相關在不同學習器下 的誤判率	98
表 5.10 三群：兩組相同共變數矩陣且負相關在不同學習器 下的誤判率	99
表 5.11 三群：共變數矩陣皆不同在不同學習器下的誤判率	100
表 5.12 三群預測資料誤判率比較	102
表 5.13 三群訓練資料誤判率比較	102
表 6.1 機械手臂隱藏層數目 vs 模型預測	109
表 7.1 相同共變數矩陣在多個學習器下的誤判率	118
表 7.2 不同共變數矩陣且正相關在多個學習器下的誤判率	119
表 7.3 不同共變數矩陣且關係一正一負在多個學習器下的誤 判率	120
表 7.4 不同共變數矩陣且負相關在多個學習器下的誤判率	122
表 7.5 不同共變數矩陣且平均數相近在多個學習器下的誤判率	124
表 7.6 相同共變數矩陣在多個學習器下的誤判率	125
表 7.7 三群資料兩組共變數矩陣相同且正相關在多個學習器 下的誤判率	127
表 7.8 三群資料兩組共變數矩陣相同且負相關在多個學習器 下的誤判率	128
表 7.9 三群資料共變數矩陣皆不同在多個學習器下的誤判率	130

序

本學期學的 L^AT_EX 與 Python 在各方面都有非常好的優勢。Python 有大量的函示庫，如 NumPy、Pandas、Matplotlib 等，使得開發者可以快速且方便地進行數據處理、機器學習、視覺化等工作。且 Python 是現今主流的程式軟體，因此廣為大眾接受，也更好與人交流。而 L^AT_EX 有專業的排版能力與數寫式子的編寫能力，在學術論文編寫上是可以達到事半功倍的效果，並且有很多不同的套件可以幫助製作文章上更多不同花樣。

本書第一章節是在簡單概要說明 L^AT_EX 的基礎操作，可以快速瞭解 L^AT_EX 專家系統的運作與排版要訣。第二章描述的是利用 Python 套件繪製圖型的方法，第三章為各種統計分配的描寫，探討統計分配的圖型與理論的證明。第四五六章講述監督式機器學習分別是迴歸模型、判別式分析與類神經網路在參數間的關係，在各種條件下探討適合的模型，並透過分群方式檢驗個字的準確率，進而進行比較。第七章為監督式機器學習的大統整，把所有模型放一起比較，更了解模型的特點與使用時機。

一整學期的磨練，真的受益良多，想剛進來研究所時什麼都不會，感謝汪教授一整學期的指導，慢慢從了解 L^AT_EX 到運用 Python 繪圖，最後接觸到機器學習，每一次作業都能讓了解到自己不足之處，進而改善更加進步。每次作業下來雖然有時會很累，但當成品做出來時又覺得離自己的目標更進一步的感覺，也更加知道要不斷的求進步，才不

會落後太多。這本書是這個學期幾個月的付出，從一開始的摸索到最後可以上手運用做成一本書，這本書可以說是努力的證明，希望未來在求學與求職上都可以運用自如，成為自己的助力。

韓明澄

2024 年 1 月於台北大學

第 1 章

L^AT_EX 基本工具手冊

¹ L^AT_EX 是一種排版系統，常用於科技論文、學術論文、書籍、演示文稿與複雜數學公式等內容。其軟體優點在於排版品質高可讓作者更專心於文檔的撰寫不需要考慮排版問題，並且有很多套件與模板可以使作者在不同條件下做出最佳的文稿，撰寫數學公式容易且輸出的品質極佳，利用 L^AT_EX 的程式碼可以輕鬆撰寫數學公式在數學、物理及工程領域都受到廣泛運用，並且 L^AT_EX 是個免費的軟體不需花任何一毛錢就可以享受專家的體驗。下文中就要簡介與模擬 L^AT_EX 的強大之處，讓每個人都能輕鬆使用 L^AT_EX 做出高品質的論文。

1.1 基本模板

開始用 L^AT_EX 前必須先瞭解好用的功能，方便作者在編輯文章時可以更快速的排版。

1.1.1 標題

文章開頭標題應用，如:`title,author,date,section,subsection,subsubsection`
使用方法如：

```
\documentclass[12pt, a4paper]{article}
```

¹此文章編輯器用的是 L^AT_EX

```
{自行定義的定義檔}
\title{ \LaTeX 補帖}
\author{{\SM 韓明澄}}
\date{{\CH \today}}
\begin{document}
\maketitle
\section{範例}
\subsection{範例}
\subsubsection{範例}
```

成果如圖 1.1。

LaTeX 補帖

韓明澄

October 2, 2023

1 範例

1.1 範例

1.1.1 範例

圖 1.1: 標題

1.1.2 文章內容的排版

文章內容常需要標號，這時不能手動編號，需使用套件來幫忙，如：

enumerate,item,itemize,equation

使用方法如：

```
\begin{enumerate}
\item $ x \in [0,1], P=1 $
\item $ x \in [-\pi, \pi], P=2\pi $
```

```
\end{enumerate}
```

成果如下

$$1. \quad x \in [0, 1], P = 1$$

$$2. \quad x \in [-\pi, \pi], P = 2\pi$$

```
\begin{enumerate}
\item\textbf{傅立葉係數:}
\begin{equation}
a_n=\frac{2}{P}\int_P s(x)\cos(2\pi x\frac{n}{P})dx
\end{equation}
\begin{equation}
b_n=\frac{2}{P}\int_P s(x)\sin(2\pi x\frac{n}{P})dx
\end{equation}
\end{enumerate}
\begin{enumerate}[resume] %[resume] 延續上段文章標碼的意思
\item\textbf{傅立葉級數，sine-cosine形式:}
\begin{equation}
s_N(x)=\frac{a_0}{2}+\sum_{n=1}^N \bigg(a_n\cos\big(\frac{2\pi nx}{P}\big)+b_n\sin\big(\frac{2\pi nx}{P}\big)\bigg)
\end{equation}
\end{enumerate}
```

成果如下

1. 傅立葉係數:

$$a_n = \frac{2}{P} \int_P s(x) \cos\left(2\pi x \frac{n}{P}\right) dx \quad (1.1)$$

$$b_n = \frac{2}{P} \int_P s(x) \sin\left(2\pi x \frac{n}{P}\right) dx \quad (1.2)$$

resume 傅立葉級數，sine-cosine 形式：

$$s_N(x) = \frac{a_0}{2} + \sum_{n=1}^N \left(a_n \cos\left(\frac{2\pi n x}{P}\right) + b_n \sin\left(\frac{2\pi n x}{P}\right) \right) \quad (1.3)$$

```
\begin{itemize}
\item $ 0 $ 代表直流信號成份
\item $\frac{-1}{8}$ 代表分頻為 $ \frac{+1}{8} $ 的信  
號強度
\item $\frac{-1}{4}$ 代表分頻為 $ \frac{+1}{4} $ 的信  
號強度
\item $\frac{-3}{8}$ 代表分頻為 $ \frac{+3}{8} $ 的信  
號強度
\item $\frac{-1}{2}$ 代表分頻為 $ \frac{+1}{2} $ 的信  
號強度
\end{itemize}
```

成果如下

- 0 代表直流信號成份
- $-\frac{1}{8}$ 代表分頻為 $\frac{+1}{8}$ 的信號強度
- $-\frac{1}{4}$ 代表分頻為 $\frac{+1}{4}$ 的信號強度
- $-\frac{3}{8}$ 代表分頻為 $\frac{+3}{8}$ 的信號強度
- $-\frac{1}{2}$ 代表分頻為 $\frac{+1}{2}$ 的信號強度

1.1.3 圖片表格標題

放圖片²時常需要標出文章對應處，如果用手動標碼容易導致標碼錯亂的狀況，因此我們需要用到 label 搭配 ref 程式碼來協助。

²要了解如何加入圖片請閱讀第四章節

使用方法如：

```
找出可能的頻率成分如(圖\ref{波頻率})。
\begin{figure}[h]
\caption{}
\centering
\includegraphics[scale=0.1]{波頻率.png}
\label{範例}
\end{figure}
```

1.1.4 註腳與程式碼樣子

當一段文章需要註腳時請引用 footnote 指令或需要呈現程式碼的樣子時使用 lstlisting。

使用方法如：

```
\begin{lstlisting}
(程式碼)
\end{lstlisting}
\footnote{(想加入的註腳)}
```

成果如下圖 1.2。

要被呈現的程式碼

¹請加入註腳

圖 1.2: 註腳與程式碼樣子

1.2 文字

LATEX 軟體中對文字有很多不同的模式可以改變，如：字型、顏色、大小...，此節來探討文字的改變。

1.2.1 字體樣式與顏色

LATEX 可以改變很多字體³樣式下面做一些範例。

黑體	儼黑 Pro
圓體	娃娃體
蘋果儼細宋	行楷
扁扁扁扁體	隸變
凌慧體	雅痞
報隸	楷體-繁

不僅只有中文可以換字體，多種語言都可以。

Times New Roman	Andale Mono
Arial	Arial Black
Charter	Academy Engraved LET
Hiragino Mincho Pro: ジ	Hiragino Kaku Gothic Pro: ジ
Apple SD Gothic Neo: 작은부분	Damascus: عربیعربی

4

使用方法如：

{想使用的字體:\MB 想輸入的文字}

1.2.2 字體大小

LATEX 中可以依照作者的想法改變字體大小方便在文中特別變化或註記

³使用不同字型時必須先定義該字體

⁴資料來源 <https://zhcnt2.ilovetranslation.com/okpYQWR3qkD=d/>

系統自訂

我們可以依照系統中幫你放大縮小：

超大字體 大號字體 小號字體 極小字體。

使用方法如：

```
\colorbox{babyblue}{{\Huge 超大字體}; {\Large 大號  
字體}; {\small 小號字體}; {\tiny 極小字體}}
```

作家自訂

不想要被系統限制也可以自行自訂想要的文字大小：

大家你好大家你好大家你好。

使用方法如：

```
{\fontsize{20pt}{20pt}\selectfont 大家你好}
```

1.2.3 字顏色

L^AT_EX 中我們可以自由選擇你喜歡字的顏色方便作者在重要的地方可以標記畫重點：

中秋快樂	中秋快樂	中秋快樂	中秋快樂	中秋快樂
中秋快樂	中秋快樂	中秋快樂	中秋快樂	中秋快樂

使用方法如：

```
\textcolor{babyblue}{想輸入文字}
```

1.2.4 結合使用

當然上述方法我們都可以結合使用讓我們的文章更多元。

白日依山盡 黃河入海流

使用方法如：

```
{\fontsize{20pt}{20pt}\selectfont{\GG\
textcolor{sapphire}{自日依山盡}}}{\fontsize
{10pt}{10pt}\selectfont{\BT\textcolor{
arylidelyellow}{黃河入海流}}}
```

1.3 數學式子

L^AT_EX 最被理工人士推崇的就是他極為方便的書寫數學式子⁵的方式，可以用簡單的幾行文字就可以寫出專業的數學公式。

1.3.1 基本運算符號

1. 加減乘除： $2 + 1; 2 - 1; 2 \times 1$ 或 $2 \cdot 1; \frac{1}{2}, 1/2, 1 \div 2$
2. 各種括號大小： $(())$, $[[[]]]$, $\{ \{ \} \}$

使用方法如：

```
加 : $ 2+1 $; 減 : $ 2-1 $; 乘 : $ 2\mathbf{\times}1$ 或 $2
\mathbf{\cdot} 1 $; 除 : $ \mathbf{\frac{1}{2}} $, $ 1/2 $, $ 1 \
\mathbf{div} 2 $  

各種括號大小:$ \mathbf{\bigg(} \mathbf{\bigg)}; \mathbf{\big(} \mathbf{\big)}; \mathbf{\big[} \mathbf{\big]}; \mathbf{\big\{} \mathbf{\big\}};
```

1.3.2 數學符號

數學上有許多專業的符號⁶，在 L^AT_EX 可以輕鬆運用。

1. 數學符號：

$$\alpha; \beta; \gamma; \delta; \epsilon; \zeta; \eta; \theta; \kappa; \mu; \lambda; \xi; \pi; \rho; \nu; \sigma; \tau; \upsilon; \phi$$

⁵使用數學符號時前後都需要加入 \$ 符號

⁶如需要大寫的數學符號只需要把開頭字母變大寫即可如:Gamma[Γ]

$\psi; \omega; \log; \equiv; \cong; \div; \pm; \mp; \in; \ni; \exists; | a |; \| a \|; \oplus; \not\oplus; \top$

使用方法如：

```
$$\alpha; \beta; \gamma; \delta; \epsilon; \zeta; \eta; \theta; \kappa; \mu; \lambda; \xi; \pi; \rho; \nu; \sigma; \tau; \upsilon; \phi; \psi; \omega; \log; \equiv; \cong; \div; \pm; \mp; \in; \ni; \exists; | a |; \| a \|; \oplus; \not\oplus; \top$$
```

% 兩個\$符號代表跳行置中

2. 數學運算符號:

$\hat{a}; \sqrt{ab}; \sqrt[n]{ab}; \log_a b; \lg ab; a^b; a_b; c_a^b; \tilde{ab}; \hat{ab}; \overrightarrow{ab}; \overbrace{ab}; \underbrace{ab}$

$\underline{ab}; \overline{ab}; \frac{\partial a}{\partial b}; \frac{dx}{dy}; \lim_{a \rightarrow b}; \int_a^b; \oint_a^b; \prod_a^b; \bigcap_a^b$

$\bigcup_a^b; \sum_a^b; \begin{pmatrix} a \\ b \end{pmatrix}; \begin{cases} a & x = 0 \\ b & x > 0 \end{cases}; \begin{bmatrix} a & b \\ c & d \end{bmatrix}$

使用方法如：

```
$$\hat{a}; \sqrt{ab}; \sqrt[n]{ab}; \log_a b; \lg ab; a^b; a_b; c_a^b; \widetilde{ab}; \widehat{ab}; \overrightarrow{ab}; \overbrace{ab}; \underbrace{ab}; \underline{ab}; \frac{\partial a}{\partial b}; \frac{dx}{dy}; \lim_{a \rightarrow b}; \int_a^b; \oint_a^b; \prod_a^b; \bigcap_a^b
```

3. 矩陣：

$$\begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \begin{bmatrix} a_1 & b_1 & c_1 & d_1 & e_1 & f_1 & g_1 \\ a_2 & b_2 & c_2 & d_2 & e_2 & f_2 & g_2 \\ a_3 & b_3 & c_3 & d_3 & e_3 & f_3 & g_3 \\ a_4 & b_4 & c_4 & d_4 & e_4 & f_4 & g_4 \\ a_5 & b_5 & c_5 & d_5 & e_5 & f_5 & g_5 \\ a_6 & b_6 & c_6 & d_6 & e_6 & f_6 & g_6 \\ a_7 & b_7 & c_7 & d_7 & e_7 & f_7 & g_7 \end{bmatrix}$$

使用方法如：

```
$ \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \begin{bmatrix} a_1 & b_1 & c_1 & d_1 & e_1 & f_1 & g_1 \\ a_2 & b_2 & c_2 & d_2 & e_2 & f_2 & g_2 \\ a_3 & b_3 & c_3 & d_3 & e_3 & f_3 & g_3 \\ a_4 & b_4 & c_4 & d_4 & e_4 & f_4 & g_4 \\ a_5 & b_5 & c_5 & d_5 & e_5 & f_5 & g_5 \\ a_6 & b_6 & c_6 & d_6 & e_6 & f_6 & g_6 \\ a_7 & b_7 & c_7 & d_7 & e_7 & f_7 & g_7 \end{bmatrix} \begin{bmatrix} a_1 & b_1 & c_1 & d_1 & e_1 & f_1 & g_1 \\ a_2 & b_2 & c_2 & d_2 & e_2 & f_2 & g_2 \\ a_3 & b_3 & c_3 & d_3 & e_3 & f_3 & g_3 \\ a_4 & b_4 & c_4 & d_4 & e_4 & f_4 & g_4 \\ a_5 & b_5 & c_5 & d_5 & e_5 & f_5 & g_5 \\ a_6 & b_6 & c_6 & d_6 & e_6 & f_6 & g_6 \\ a_7 & b_7 & c_7 & d_7 & e_7 & f_7 & g_7 \end{bmatrix}
```

4. 複雜數學式子示範⁷：

$$\bullet \mathbf{x} = (\mathbf{v}_1^T \mathbf{x}) \mathbf{v}_1 + (\mathbf{v}_2^T \mathbf{x}) \mathbf{v}_2 + (\mathbf{v}_3^T \mathbf{x}) \mathbf{v}_3 = -\frac{1}{2} \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} + \frac{4}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \frac{1}{6} \begin{bmatrix} 1 \\ 1 \\ -2 \end{bmatrix}$$

⁷將基礎運算符號組合在一起就可以讓數學式子變複雜

- $a_k = \langle \cos(kx), f(x) \rangle = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(kx) dx, k = 0, 1, \dots, n$

- $W = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \omega^{3(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix},$

- $\widehat{s}_n \triangleq \begin{cases} \frac{A_0}{2} & = \frac{a_0}{2} & n = 0 \\ \frac{A_n}{2} e^{-i\varphi_n} & = \frac{1}{2}(a_n - ib_n) & n > 0 \\ \widehat{s}_{|n|}^* & & n < 0 \end{cases}$

- $A_n \cos\left(\frac{2\pi nx}{P} - \varphi_n\right) \equiv \underbrace{A_n \cos(\varphi_n)}_{a_n} \cos\left(\frac{2\pi nx}{P}\right) + \underbrace{A_n \sin(\varphi_n)}_{b_n} \sin\left(\frac{2\pi nx}{P}\right)$

1.4 圖片

對文章來說圖片也是必不可少元素，加入圖片不僅可以增加文章的豐富度更可以讓讀者更瞭解作者的思想，L^AT_EX 也可以放入多種類型圖片，不僅如此還可以更改圖片的大小角度還有呈現的模式。

1.4.1 加入圖片

匯入圖片如圖 1.4 與圖 1.3：



圖 1.3: 匯入 pdf 檔圖



圖 1.4: 匯入 PNG 檔圖

可以等比例調整圖片大小⁸如圖 1.5，也可以讓圖片選轉角度如圖 1.6：



圖 1.5: 調整大小

⁸調整圖的大小也可以用 width 代替 scale 做調整



圖 1.6: 旋轉角度

使用方法如：

```
\begin{figure}[h]
\centering % 置中
\includegraphics[圖角度:angle=15, 圖大小:scale
=0.1]{檔案名:\imgdir Petals.png}
\caption{圖名}
\label{標籤}
\end{figure}\\"
```

1.4.2 多張圖片並排

兩張圖片並排如圖 1.7：



(a) 可愛貓

(b) 超可愛貓

圖 1.7: 兩圖並排

當然也可以多張圖並排如圖 1.8：



(a) 可愛貓

(b) 超可愛貓

(c) 無敵可愛貓

圖 1.8: 三圖並排

使用方法如：

```
\subfloat[可愛貓]{
\includegraphics[scale=0.2]{\imgdir 貓1.jpg}}
\subfloat[超可愛貓]{
\includegraphics[scale=0.2]{\imgdir 貓2.jpg}}
\subfloat[無敵可愛貓]{
\includegraphics[scale=0.2]{\imgdir 貓3.jpg}}
```

也可以將圖排成四宮格如圖 1.9：



(a) 可愛貓



(b) 超可愛貓



(c) 無敵可愛貓



(d) 一般貓

圖 1.9: 四宮格排圖

1.4.3 文字旁的子圖

日本建築擁有久遠的歷史，最早在 6 世紀受到中國建築的影響，佛教在傳入日本時同時帶入中國隋唐的建築技術與風格，大量興建佛寺和宮殿，隨後慢慢發展出屬於日本的獨特風格。自 16 世紀起，府邸和城樓取代佛寺成為主要建築活動，城堡在江戶時代已演變為地方的政治與經濟中心，各大城市的富有人家也開始興建不同規模與風格的府邸。

日本在明治時代開始引入西方的建築技巧、材料和風格，建造與傳統風格有極大差別的鋼鐵和水泥建築。

二戰前最早的日本建築受到西方影響，材料、功能、結構和比例之間的關係得到最大限度的發展，二戰後，重建的城市與舊有的建築樣貌完全不同，逐而發展出現代的日本城市建築風格。同一時期，日本經濟快速成長，許多城市建築（例如東京鐵塔）都是在 20 世紀中後期興建的，當時是日本建築粗野現代主義建築的最興盛時期，代表建築包括在 1961 年落成的東京文化會館。



使用方法如：

```
\begin{minipage}{.5\linewidth}
{內容}
\end{minipage}
\begin{minipage}{.6\linewidth}
\centering
\includegraphics[scale=0.18]{\imgdir 日本1.jpg}
\end{minipage}
```

1.5 表格

做數理研究免不了需要很多數據，因此表格是必不可少的工具，而在 L^AT_EX 中也可以輕鬆按照自己的想法製造出最理想的表格。

1.5.1 基礎表格

最簡單 2×2 表格如表 1.1。

表 1.1: 2×2 表格

abc	def
ghi	jkł

使用方法如：

```
\begin{table}[h]
\centering
\caption{$ 2\times2 $表格}
\label{$ 2\times2 $表格}
\extrarowheight=5pt
\begin{tabular}{c|c}
abc & def\\ \hline
ghi & jkl\\
\end{tabular}
\end{table}
```

表 1.2: 有瑕疵的基本的表格

車款	台灣銷售量(輛)		2021 年	2022 年
Toyota Yaris			1,247,585	168,557
Toyota Corolla			110824	131,548
Nissan Note			90183	110,113

以上資料來自 yahoo 新聞。

但我們發現表 1.2 左上方的地方會因為字的多寡而變得不美觀，因此我們可以利用別種方法來製作表格如表 1.3，當然表格也能旋轉如表 1.4：使用方法如：

表 1.3: 基本的表格

車款	台灣銷售量(輛)	
	2021 年	2022 年
Toyota Yaris	1,247,585	168,557
Toyota Corolla	110824	131,548
Nissan Note	90183	110,113

```
\extrarowheight=5pt
\begin{tabular}{lrr}
& \multicolumn{2}{c}{台灣銷售量(輛)} \\
\cline{2-3}
車款 & 2021年 & 2022年 \\ \hline
Toyota Yaris & 1,247,585 & 168,557 \\
Toyota Corolla & 110824 & 131,548 \\
Nissan Note & 90183 & 110,113 \\
\hline
\end{tabular}
```

表 1.4: 旋轉表格

車款	台灣銷售量(輛)	
	2021 年	2022 年
Toyota Yaris	1,247,585	168,557
Toyota Corolla	110824	131,548
Nissan Note	90183	110,113

使用方法如：

```
\rotatebox[origin=c]{90}{
\begin{tabular}{lrr}
```

表格中也可指定特定的區塊如表 1.5、表 1.6、行如表 1.7 或列如表 1.8 來填上色塊標註重點⁹：全表格填色使用方法如：

表 1.5: 全表格填色

車款	台灣銷售量(輛)	
	2021 年	2022 年
Toyota Yaris	1,247,585	168,557
Toyota Corolla	110824	131,548
Nissan Note	90183	110,113

```
\colorbox{babyblue}{
\begin{tabular}{lrr}
表格內容
}
```

⁹對表格填色時有時候顏色太深會影響表格的能見度，可去自訂義中調整透明度

表 1.6: 表格指定區域填色

車款	台灣銷售量 (輛)	
	2021 年	2022 年
Toyota Yaris	1,247,585	168,557
Toyota Corolla	110824	131,548
Nissan Note	90183	110,113

指定區域填色使用方法如:

```
\cellcolor{babyblue}Toyota Yaris & 1,247,585
& 168,557 \\
Toyota Corolla & 110824 \cellcolor{
seagreen}& 131,548\\
Nissan Note & 90183 & \cellcolor{
limegreen}110,113 \\ \hline
```

行填色使用方法如:

```
\begin{tabular}{>{\columncolor{arylideyellow}}l r
>{\columncolor{slight}}r}
```

表 1.7: 表格行填色

車款	台灣銷售量 (輛)	
	2021 年	2022 年
Toyota Yaris	1,247,585	168,557
Toyota Corolla	110824	131,548
Nissan Note	90183	110,113

列填色使用方法如:

```
\rowcolor{gray(x11gray)}Toyota Yaris &
1,247,585 & 168,557 \\
```

表 1.8: 表格列填色

台灣銷售量(輛)		
車款	2021 年	2022 年
Toyota Yaris	1,247,585	168,557
Toyota Corolla	110824	131,548
Nissan Note	90183	110,113

1.5.2 特別表格

- 並排表格：可將兩個表格並排放在一起方便交互比較如表 1.9 。

使用方法如：

```
\renewcommand{\arraystretch}{1.8}
\extrarowheight=4pt
\colorbox{babyblue}{\begin{tabular}{lrr}
& \multicolumn{2}{c}{台灣銷售量(輛)} \\
\cline{2-3}
車款 & 2021年 & 2022年 \\ \hline
Toyota Yaris & 1,247,585 & 168,557 \\
Toyota Corolla & 110824 & 131,548 \\
Nissan Note & 90183 & 110,113 \\
\\ \hline
\end{tabular}}\hspace{10pt}
\begin{tabular}{lrr}
& \multicolumn{2}{c}{價格(NT)} \\
\cline{2-3}
車款 & 2021年 & 2022年 \\ \hline
\rowcolor{limegreen} Toyota Yaris & 70W \\
& 86W \\
\rowcolor{sapphire} Toyota Corolla & 100W & 90W \\
Nissan Note & 63W & 66W \\
\end{tabular}
```

- 粗細：利用 booktabs 套件改變表格線的粗細如表 1.10 ,booktabs 套

表 1.9: 兩個表格並列擺放

台灣銷售量 (輛)		
車款	2021 年	2022 年
Toyota Yaris	1,247,585	168,557
Toyota Corolla	110824	131,548
Nissan Note	90183	110,113

價格 (NT)		
車款	2021 年	2022 年
Toyota Yaris	70W	86W
Toyota Corolla	100W	90W
Nissan Note	63W	66W

件提供 `toprule`、`midrule` 和 `bottomrule` 指令來幫表格線條加粗¹⁰

- 利用 `booktabs` 也可以用 `addlinespace` 調整垂直間距如表 1.11。

¹⁰套件要到自己的定義檔中加入 `\usepackage{booktabs}` 新增套件

表 1.10: booktabs 套件改變表格格線粗細

Source	SS	df	MS	
M odel	3089.06361	1	3089.06361	
Residual	33043.769	17565	1.88122795	
Total	36132.8326	17566	2.05697555	
children	Coef.	Std.Err	t	p> t
schooling	-0.0959731	0.0023684	-40.52	0.000
cons	3.161674	0.016408	192.69	0.000

表 1.11: booktabs 套件改變垂直間距

Source	SS	df	MS	
M odel	3089.06361	1	3089.06361	
Residual	33043.769	17565	1.88122795	
Total	36132.8326	17566	2.05697555	
children	Coef.	Std.Err	t	p> t
schooling	-0.0959731	0.0023684	-40.52	0.000
cons	3.161674	0.016408	192.69	0.000

使用 bookstabs 套件時因為加深水平線會導致垂直線會有斷線的情況如表 1.12，要盡量避免使用該套件時加入垂直線。

表 1.12: booktabs 斷線問題

Source	SS	df	MS
Model	3089.06361	1	3089.06361
Residual	33043.769	17565	1.88122795
Total	36132.8326	17566	2.05697555
	Coef.	Std.Err	t
children	-0.0959731	0.0023684	-40.52
schooling	3.161674	0.016408	192.69
cons			0.000

4. 文字旁加表格當表格需要一段文字說明時，可以使用 minipage 來呈現如下：

自 2019 年開始的全球新冠疫情，以及 2022 年爆發的烏俄戰爭影響下，導致 2022 年出現全球性通貨膨脹，再加上先前已發生的全球汽車產能不足、車用晶片短缺與原物料上漲壓力，終究反應在新車成本增加與普遍性的缺車，這也直接考驗各車廠對於新車產能應對的能力。2022 年臺灣汽車市場就在產能受限

車款	台灣銷售量(輛)	
	2021 年	2022 年
Toyota Yaris	1,247,585	168,557
Toyota Corolla	110824	131,548
Nissan Note	90183	110,113

與成本上漲等壓力下，總市場全年度最終繳出 429,731 輛，相較於 2021 年出現 4.5% 的衰退幅度，各大車廠多數處於銷量萎縮的格局。

5. 利用 excel 做表格在做一些資料較多的表格時, 用上述方法來做很容易做到亂掉, 這時可以利用 excel 搭配巨集程式 Excel2LaTeX 來搞定這些比較麻煩的表格, 做出來如圖 1.10。

	A	B	C	D	E	F
1	name	age	high(cm)	weight(kg)	score	
2	Ben	11	143.6	45.3	78	
3	Jack	34	180	93.4	34	
4	Leo	23	172.9	58.4	95	
5	Lisa	45	154.6	45.1	75	
6	Rose	13	130.6	39.2	39	
7	Wandy	25	165.7	78	85	
8	Tom	37	198.4	102.6	64	
9	Tommy	67	164.3	54.2	73	
10	Guy	20	145.9	39.3	26	

(a) excel 圖的表

name	age	high(cm)	weight(kg)
Ben	11	143.6	45.3
Jack	34	180	93.4
Leo	23	172.9	58.4
Lisa	45	154.6	45.1
Rose	13	130.6	39.2
Wandy	25	165.7	78
Tom	37	198.4	102.6
Tommy	67	164.3	54.2
Guy	20	145.9	39.3

(b) excel 做 latex 表

圖 1.10: excel 做 latex 表

6. 長表格運用遇到多比資料, 有時會超出頁面限制, 這時可用長表格來接續呈現如表 1.13。

表 1.13: 長型表格

NAME	MIN	PTS	FG%	3P%	FT%	REB	AST
Joel	34.6	33.1	54.8	33	85.7	10.2	4.2
Luka	36.2	32.4	49.6	34.2	74.2	8.6	8
Damian	36.3	32.2	46.3	37.1	91.4	4.8	7.3
Shai	35.5	31.4	51	34.5	90.5	4.8	5.5
Giannis	32.1	31.1	55.3	27.5	64.5	11.8	5.7
Jayson	36.9	30.1	46.6	35	85.4	8.8	4.6
Stephen	34.7	29.4	49.3	42.7	91.5	6.1	6.3
Kevin	35.6	29.1	56	40.4	91.9	6.7	5

續接下頁

承接上頁

NAME	MIN	PTS	FG%	3P%	FT%	REB	AST
LeBron	35.5	28.9	50	32.1	76.8	8.3	6.8
Donovan	35.8	28.3	48.4	38.6	86.7	4.3	4.4
Devin	34.6	27.8	49.4	35.1	85.5	4.5	5.5
Kyrie	37.4	27.1	49.4	37.9	90.5	5.1	5.5
Jaylen	35.9	26.6	49.1	33.5	76.5	6.9	3.5
Trae	34.8	26.2	43	33.5	88.6	3	
Ja	31.9	26.2	46.6	30.7	74.8	5.9	8.1
Zion	33	26	60.8	36.8	71.4	7	4.6
Anthony	34	25.9	56.3	25.7	78.4	12.5	2.6
Lauri	34.4	25.6	49.9	39.2	87.5	8.6	1.9
Julius	35.5	25.1	46	34.3	75.7	10	4.1
De'Aaron	33.4	25	51.2	32.4	78	4.2	6.1
Zach	35.9	24.8	48.5	37.5	84.8	4.5	4.2
Brandon	34.2	24.7	48.4	39	88.2	5.5	5.8
Anthony	36	24.6	45.9	36.9	75.6	5.8	4.4
DeMar	36.2	24.5	50.4	32.4	87.2	4.6	5.1
Nikola	33.7	24.5	63.2	38.3	82.2	11.8	9.8
Pascal	37.4	24.2	48	32.4	77.4	7.8	5.8
Jalen	35	24	49.1	41.6	82.9	3.5	6.2
Kawhi	33.6	23.8	51.2	41.6	87.1	6.5	3.9
Paul	34.6	23.8	45.7	37.1	87.1	6.1	5.1
LaMelo	35.2	23.3	41.1	37.6	84.3	6.4	8.4
Bradley	33.5	23.2	50.6	36.5	84.2	3.9	5.4

續接下頁

承接上頁

NAME	MIN	PTS	FG%	3P%	FT%	REB	AST
Kristaps	32.6	23.2	49.8	38.5	85.1	8.4	2.7
Jimmy	33.4	22.9	53.9	35	85	5.9	5.3
Jalen	34.2	22.1	41.7	33.8	78.6	3.7	3.7
Keldon	32.7	22	45.2	32.9	74.9	5	2.9

使用方法如：

```
\begin{longtable}{@{}lrrrrrrr@{}}
\caption{長型表格}
\label{longtable} \\
\toprule
{表格最上一列內容} \\
\midrule
\endfirsthead
\multicolumn{6}{l}{\{承接上頁\}} \\
\toprule
{表格最上一列內容} \\
\midrule
\endhead
\midrule
\multicolumn{8}{r}{\{續接下頁\}} \\
\endfoot
\endlastfoot
{表格中內容}
\end{longtable}
```

1.6 結論

上述的 L^AT_EX 使用方法都是在以基本操作為前提下都適用，當然這些操作只是 L^AT_EX 軟體中的冰山一角，還有很多進階的操作需要我們去發掘加以應用，畢竟知道的越多做起事來更省力，運用 L^AT_EX 可以使你的排版

更加靈活快速，雖然剛開始對於操作上有許多不熟悉，但只要多加練習，很快就能上手並運用自如了。

第 2 章

Python 的數學製圖

在 Python 中，數學製圖是一個主要的主題，它通過強大的繪圖庫如 Matplotlib 和 Seaborn，使數學和科學可視化變得輕鬆。這些庫提供了各種功能，讓用戶能夠創建精美的圖形，包括散點圖、折線圖、長條圖和熱圖等，這些圖形可用於呈現數據、函數、方程式和數學模型的可視化。通過 Matplotlib 的強大功能，我們可以自定義圖形的外觀，添加標籤、標題和軸標籤，使圖形變得更具信息價值。此外，Seaborn 提供了高級統計圖形和配色方案，使數據分析更容易，本文會著重在 Matplotlib 的運用上。

2.1 Matplotlib 函數圖形的繪製

本節將有十一題數學問題，將運用 Matplotlib 與 numpy 套件來繪製函數圖形，並運用圖形來解答數學問題。

2.1.1 特殊三角函數與極限

圖 2.1 將以子圖的方式繪製

$$f(x) = \frac{\sin(x)}{x}, g(x) = \frac{\sin(x^2)}{x}, h(x) = \frac{\sin^2(2x)}{x^2}$$

, 將三個函式分別觀察並加入 $x = 0$ 並標註出交點觀察極限值的收斂。

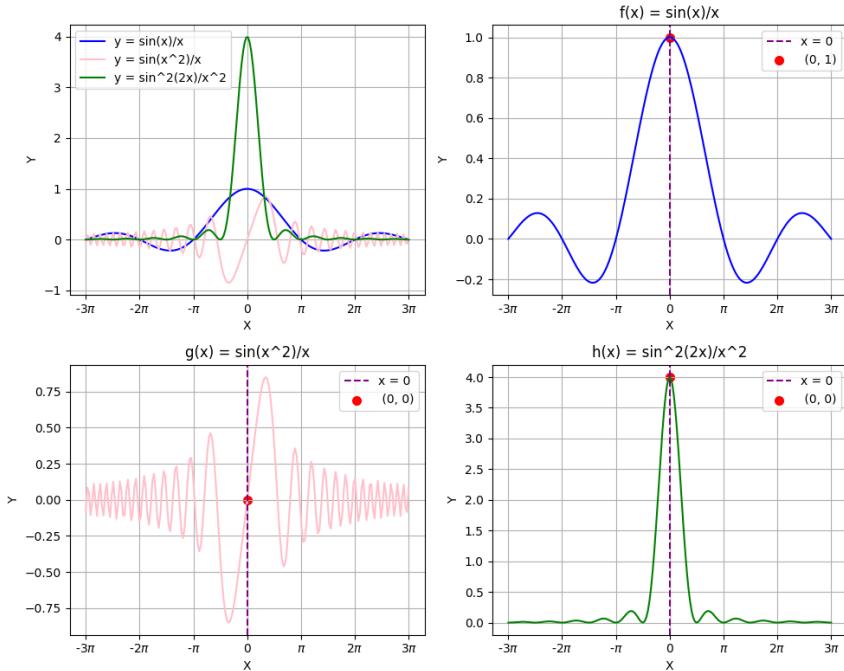


圖 2.1: $f(x) = \frac{\sin(x)}{x}$, $g(x) = \frac{\sin(x^2)}{x}$ 與 $h(x) = \frac{\sin^2(2x)}{x^2}$

根據圖 2.1 得知 $\frac{\sin(x)}{x}$ 在 x 趨近於 0 時會極限會收斂於 1 , $\frac{\sin(x^2)}{x}$ 在 x 趨近於 0 時會極限會收斂於 0 , $\frac{\sin^2(2x)}{x^2}$ 在 x 趨近於 0 時會極限會收斂於 4 。製作其中一個子圖程式碼:

```
fig, axs = plt.subplots(2, 2, figsize=(10, 8)) #  
    創建一個2x2子圖  
axs[0, 0].plot(x, y1, label='y = sin(x)/x', color  
                ='blue')  
axs[0, 0].plot(x, y2, label='y = sin(x^2)/x',  
                color='pink')  
axs[0, 0].plot(x, y3, label='y = sin^2(2x)/x^2',  
                color='green')  
axs[0, 0].grid(True)  
axs[0, 0].set_xlabel('X')  
axs[0, 0].set_ylabel('Y')
```

```

axs[0, 0].legend(loc='upper left', bbox_to_anchor
=(1, 1))
axs[0, 0].set_title('$f(x) = \frac{\sin(x)}{x}$\n$g(x) = \frac{\sin(x^2)}{x}$\n$h(x) = \frac{\sin^2(2x)}{x}$')

```

2.1.2 函式在不同 α 值下的表現

圖 2.2 嘗試在不同 α 值下，函式：

$$\frac{e^{\alpha x}}{e^{\alpha x} + 1}$$

會如何變化，並加入上下漸近線，將其放在一起觀察。因此我們可以得知在 α 越大的情況下，函數會越快往漸近線靠攏，並且不管 α 值如何他們都有共同的反曲點 $(0, 0.5)$ 。

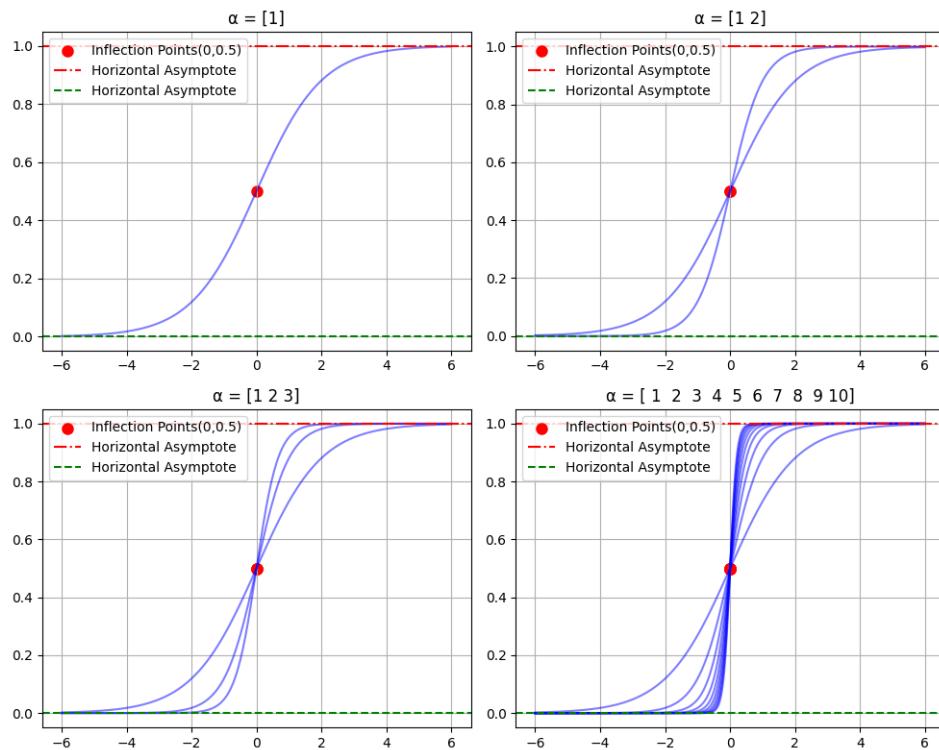


圖 2.2: $\frac{e^{\alpha x}}{e^{\alpha x} + 1}$ 與 α 關係圖

製作不同 α 值所成的圖程式碼：¹

```
def derivative(alpha, x):
    return alpha * np.exp(alpha * x) / (np.exp
        (alpha * x) + 1)**2
x_zeros = []
y_zeros = []
for a in alpha[:i + 1]:
    sol = fsolve(derivative, 0, args=(a,))
    x_zeros.append(sol[0])
    y_zeros.append(np.exp(sol) / (np.exp(sol)
        + 1))
axes[row, col].scatter(x_zeros, y_zeros, color
    ='red', marker='o', lw=3, label='Inflection
    Points(0,0.5)')
axes[row, col].axhline(y=horizontal_asymptote
    1, color='red', linestyle='-.', label=f'
    Horizontal Asymptote')
axes[row, col].axhline(y=horizontal_asymptote
    2, color='red', linestyle='--', label=f'
    Horizontal Asymptote')
axes[row, col].legend()
```

製作反曲點程式碼：²

```
def derivative(alpha, x):
    return alpha * np.exp(alpha * x) / (np.exp
        (alpha * x) + 1)**2
x_zeros = []
y_zeros = []
for a in alpha_new:
    sol = fsolve(derivative, 0, args=(a,))
    x_zeros.append(sol[0])
    y_zeros.append(np.exp(sol) / (np.exp(sol)
        + 1))
```

根據圖 2.3 能得知函式有上下兩條漸近線，分別是 $y = 1$ 與 $y = 0$ ，且不

¹ 程式碼第一行只能使用 `range(4)`，在設定新子圖覆蓋最後一張的圖，不能使用 `range(3)`，不然在後一張子圖會無法顯示。

² 反曲點就是導數為零的點，因此使用 `fsolve` 來尋找

同的 α 值都是一樣的漸近線。

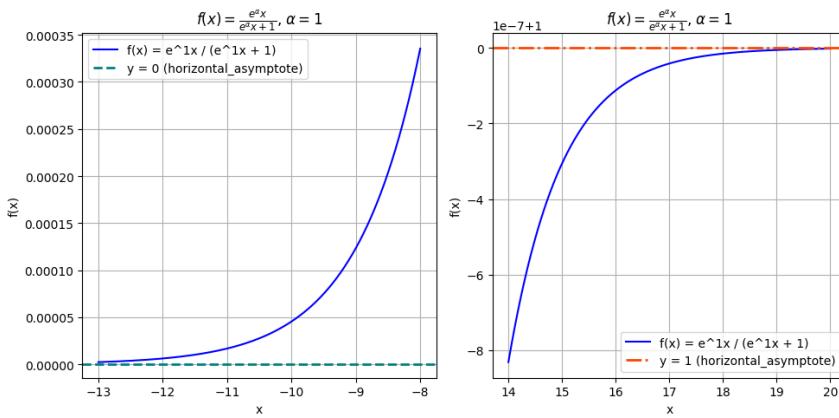


圖 2.3: $\frac{e^{\alpha x}}{e^{\alpha x} + 1}$ 與其漸近線

製作漸近線程式碼：

```
plt.subplot(1, 2, 1)
x1 = np.linspace(14, 20, 1000)
y1 = f(x1, alpha)
horizontal_asymptote1 = 1
plt.plot(x1, y1, label=f'f(x) = e^{alpha}x / (e^{alpha}x + 1)', color='blue')
plt.axhline(y=horizontal_asymptote1, color='FF
    4500', linestyle='-.', lw=2, label=f'y = {
        horizontal_asymptote1} (horizontal_asymptote)')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('$f(x) = \frac{e^{\alpha x}}{e^{\alpha x} + 1}$, $\alpha=1$')
```

2.1.3 函式震盪幅度

圖 2.4 探討函式 $f(x) = e^{-\frac{x}{10}} \sin(x)$ 在 x 值增加的情況下，震盪的幅度會有怎樣的變化。在此將函式範圍分別設定在 0 到 20、20 到 40、40 到 60 與 0 到 100 四組，並且找出他的遞減函式（Damping Trend） $e^{-\frac{x}{10}}$ 來做比較。

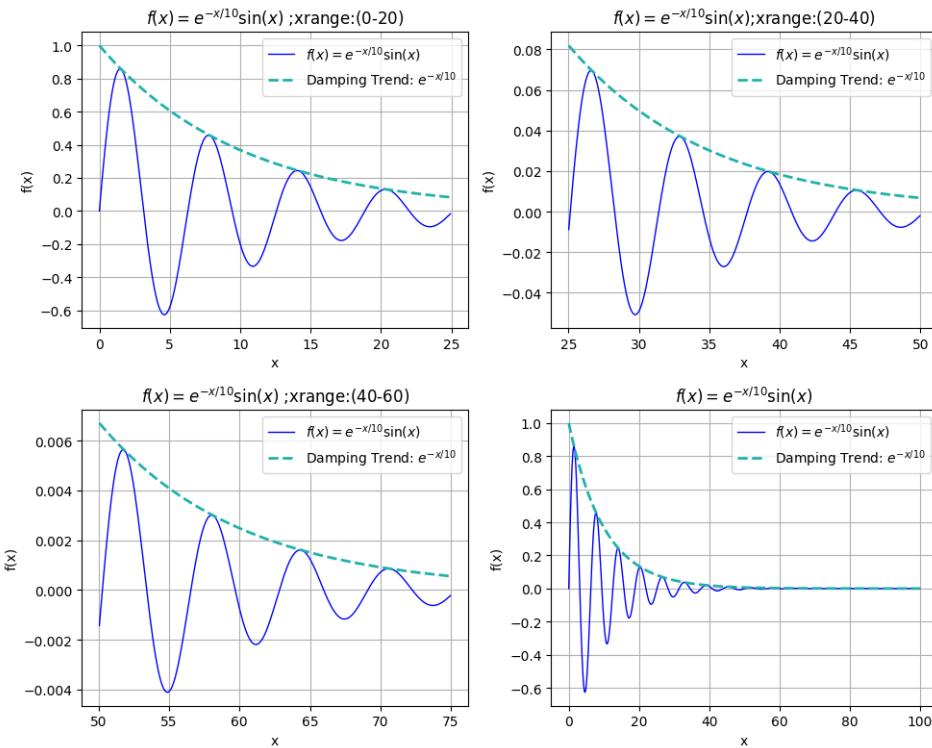


圖 2.4: $f(x) = e^{-x/10} \sin(x)$ 震盪幅度

圖 2.4 顯示當 x 值越大函式震盪幅度會隨著遞減，且每單位遞減的幅度都相同。

設定範圍製作子圖程式碼：³

```
# 第一個子圖 : 0 到 20
plt.subplot(2, 2, 1)
plt.plot(x[:250], y[:250], label='f(x) = e^{-x/10} \sin(x)', lw=1, color='blue')
plt.plot(x[:250], damping_trend[:250], label='Damping Trend: e^{-x/10}', color="#20B2AA", lw=2, linestyle='--')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('f(x) = e^{-x/10} \sin(x) ; xrange : (0-20)')
```

³ $x[:250], y[:250]$ 是在 0-250 元素裡索引需要的範圍。

```
plt.grid(True)  
plt.legend()
```

2.1.4 函數值無定義

圖 2.5為函式 $f(x) = \frac{1}{x-2}$ ，此函式在 $x = 2$ 時無定義，並製作了該函式的漸近線 $y = 0$ 如圖 2.6。

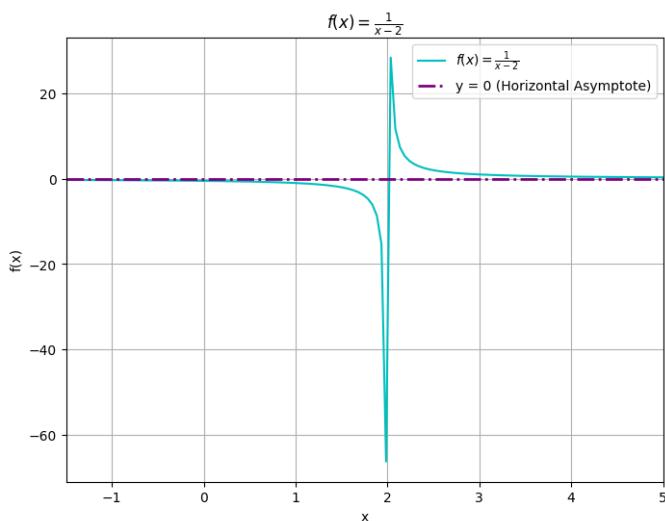
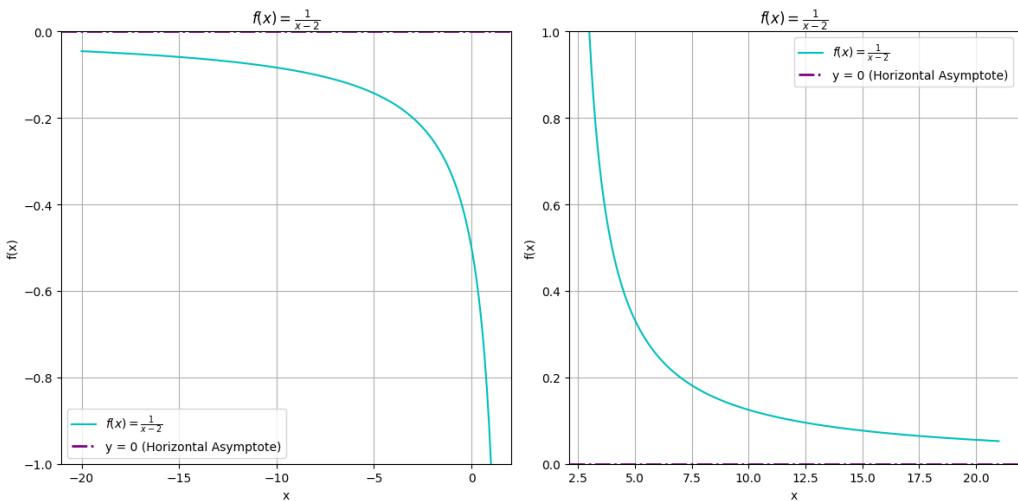


圖 2.5: $f(x) = \frac{1}{x-2}$

圖 2.6: $f(x) = \frac{1}{x-2}$ 漸近線

處理分母為零的情況程式碼：

```
x = np.setdiff1d(np.linspace(-5, 5, 200), [2])
```

2.1.5 函數、反函數與對稱線

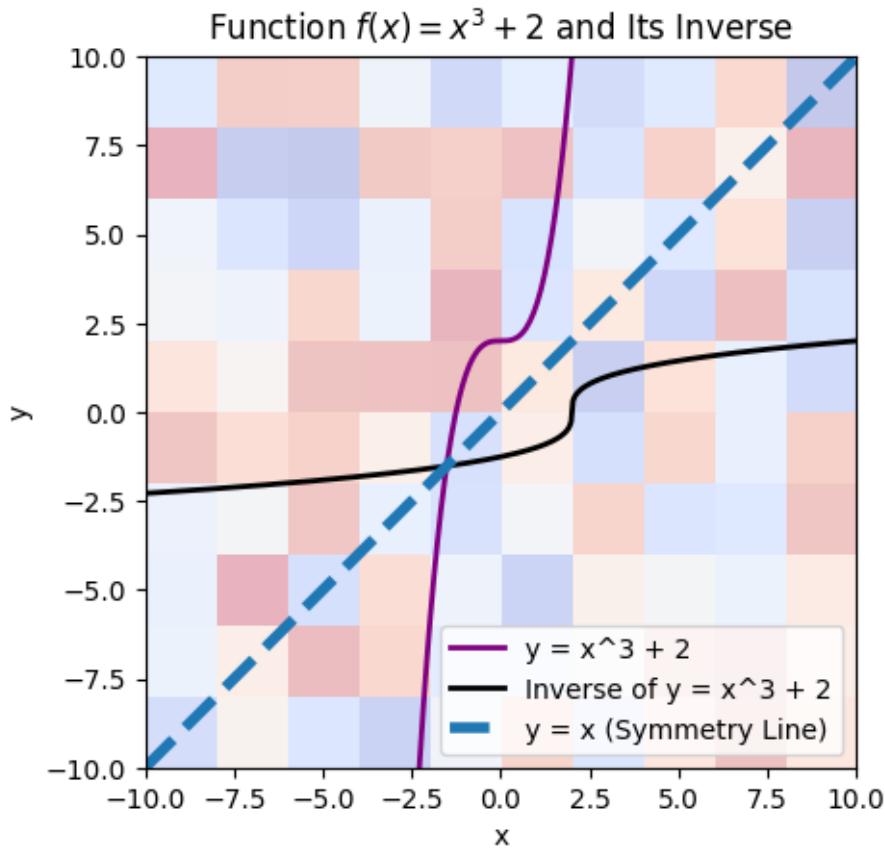
圖 2.7 在探討函數 $f(x) = x^3 + 2$ 與反函數 $(x - 2)^{\frac{1}{3}}$ ，加入兩者的對稱線 $x = y$ 方便觀察，並且改變網格顏色。利用 numpy 套件製作反函數程式碼⁴:

```
def f_inv(x):
    return np.cbrt(x - 2)
```

製作彩色網格程式碼：

```
plt.imshow(np.random.random((10, 10)), extent
           =[-10, 10, -10, 10], cmap='coolwarm', alpha
           =0.3)
```

⁴numpy 在一般情況下無法計算負數開立方，因此使用 np.cbrt 來解決

圖 2.7: $f(x) = x^3 + 2$ 與其反函數

2.1.6 常態分配

圖 2.8為函式 $f(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{(x-3)^2}{2}}$ 是一個 $\mu = 3$ ， $\sigma = 1$ 的常態分配函數，再利用 68–95–99.7 法則來區分距平均值小於一個標準差、二個標準差與三個標準差以內的百分比，並填上顏色方便區別。

68–95–99.7 法則顏色區分程式碼：

```
sigma = 1 # 標準差
mu = 3 # 平均值
x1 = np.linspace(mu - 3*sigma, mu + 3*sigma, 1000)
y1 = f(x1)
plt.fill_between(x1, y1, color='green', alpha=0.5,
                 label='68%')
```

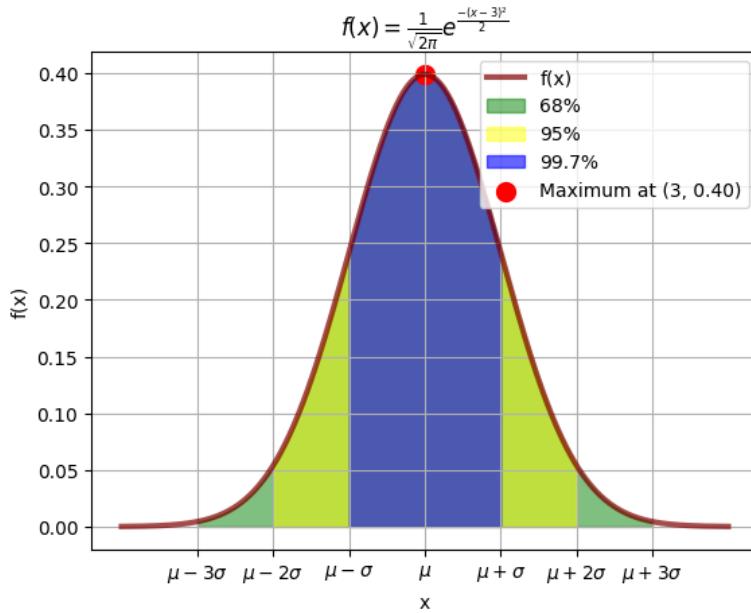


圖 2.8: $f(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{(x-3)^2}{2}}$ 與 68–95–99.7 法則

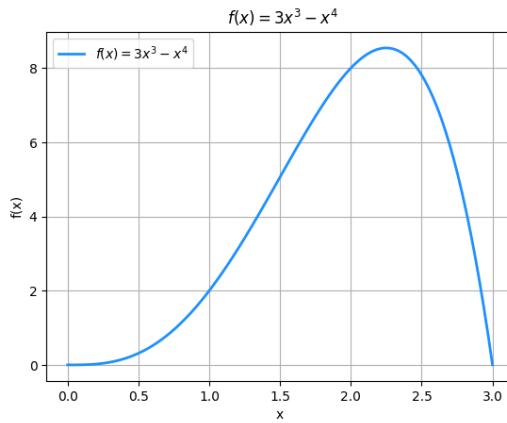
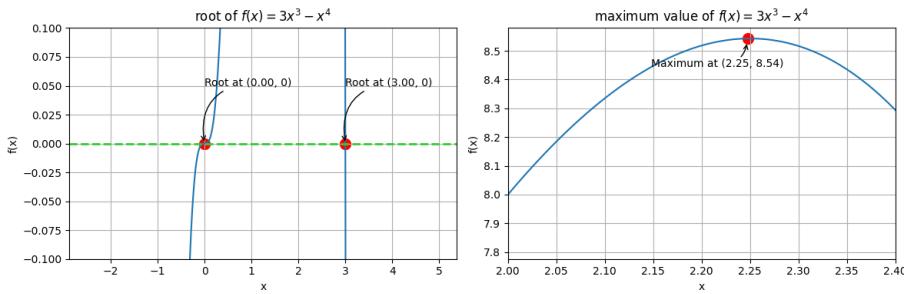
```

x2 = np.linspace(mu - 2*sigma, mu + 2*sigma, 1000)
y2 = f(x2)
plt.fill_between(x2, y2, color='yellow', alpha=0.5, label='95%')
x3 = np.linspace(mu - sigma, mu + sigma, 1000)
y3 = f(x3)
plt.fill_between(x3, y3, color='blue', alpha=0.6, label='99.7%')

```

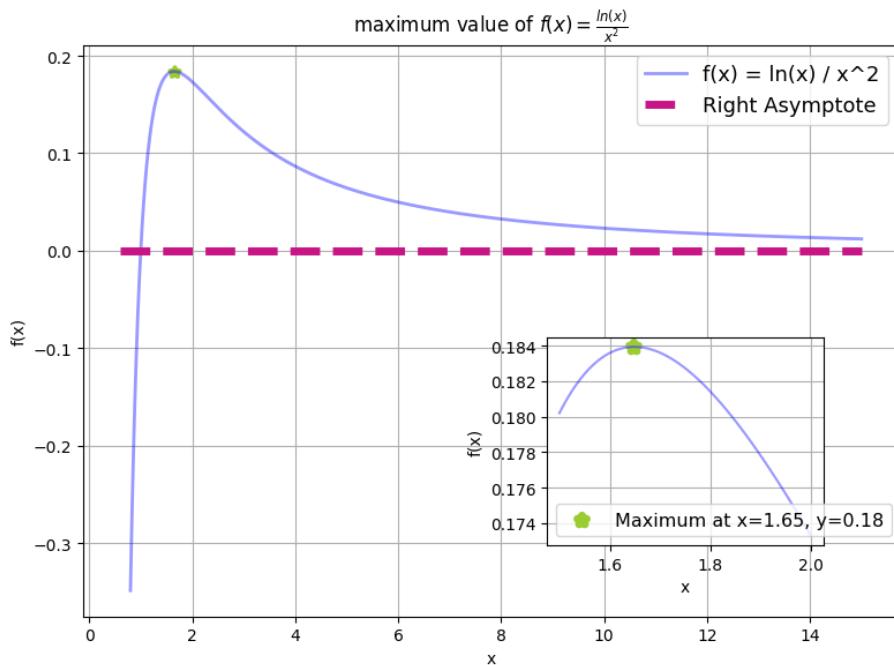
2.1.7 函數找根值

圖 2.10為函數 $f(x) = 3x^3 - x^4$ 如圖 2.9，接下將其繪製 $y = 0$ 找交點，我們得到函數有兩個實數根分別為 0 與 3，並且函數最大值為 8.543。

圖 2.9: $f(x) = 3x^3 - x^4$ 的實數根與最大值圖 2.10: $f(x) = 3x^3 - x^4$ 的實數根與最大值

2.1.8 函數的右漸近線與最大值

圖 2.11 為函數 $f(x) = \frac{\ln x}{x^2}$ ，我以附圖的方式放大函數最大值的部分為 0.18，且放大右漸近線 $y = 0$ ，來方便觀察與函數的趨勢，得知當 x 值越大函數往 $y = 0$ 漸近。

圖 2.11: $f(x) = \frac{\ln x}{x^2}$ 右漸近線與最大值

在右下放副圖程式碼：

```

ax2 = fig.add_axes([0.6, 0.2, 0.3, 0.3]) # 小圖的
    位置和大小
# 找到最大值的位置
max_index = np.argmax(y_original)
max_x = x_original[max_index]
max_y = y_original[max_index]
# 繪製函數圖形在ax2上
ax2.plot(x_original, y_original, color='blue',
          alpha=0.4)
ax2.scatter(max_x, max_y, color='#9ACD32', lw=5,
            label=f'Maximum at x={max_x:.2f}, y={max_y:.2f}',
            marker='*')
ax2.set_xlabel('x')
ax2.set_ylabel('f(x)')
ax2.legend(fontsize=11.5)
ax2.grid(True)

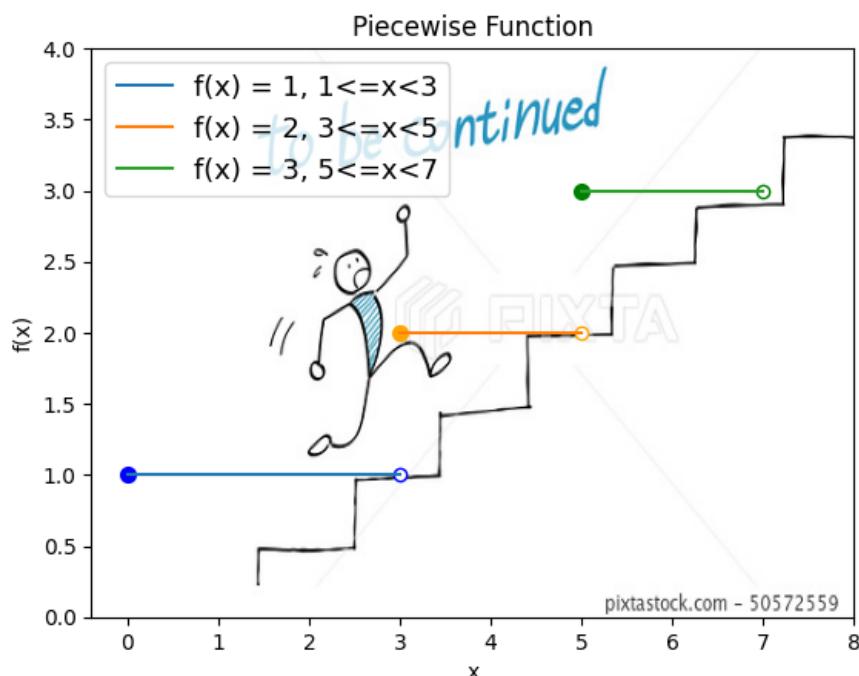
```

2.1.9 階梯函數

圖 2.12 為階梯函數：

$$f(x) = \begin{cases} 1, & 1 \leq x < 3 \\ 2, & 3 \leq x < 5, \\ 3, & 5 \leq x < 7 \end{cases}$$

有等號的端點用實心圓代表，反之就用空心圓代表，並加入一個階梯圖美化函數。



$$\text{圖 2.12: } f(x) = \begin{cases} 1, & 1 \leq x < 3 \\ 2, & 3 \leq x < 5 \\ 3, & 5 \leq x < 7 \end{cases}$$

製作端點程式碼：

```
ax.scatter(0, 1, color='b', lw=2, marker='o')
```

```

ax.scatter(3, 1, color='white', edgecolors='b',
           marker='o')
# f(x)=2
ax.scatter(3, 2, color='orange', lw=2, marker='o')
ax.scatter(5, 2, color='white', edgecolors='orange',
           marker='o')
# f(x)=3
ax.scatter(5, 3, color='green', lw=2, marker='o')
ax.scatter(7, 3, color='white', edgecolors='green',
           marker='o', )

```

更換背景圖程式碼：

```

background_image = plt.imread('/Users/hanmingcheng
    /Documents/images/stair.jpg')
ax.imshow(background_image, extent=[1, 8, 0, 4],
           aspect='auto')

```

2.1.10 單位圓

1. 變數法

圖 2.13 是利用設定 $x = \sin\theta, y = \cos\theta, 0 < 0 \leq 2\pi$ 變數來製作單位圓 $x^2 + y^2 = 1$ ，且改為用十字座標軸來展現。

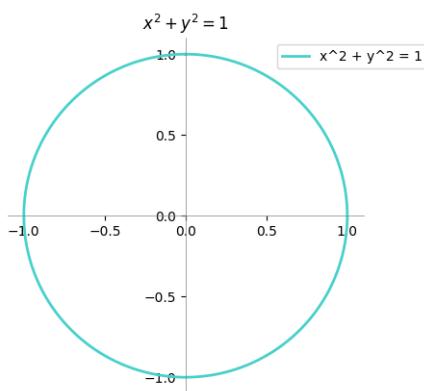


圖 2.13: 變數法製作單位圓

變數法單位圓程式碼：

```
theta = np.linspace(0, 2 * np.pi, 1000)
# 使用 x 和 y 的函數表示
x = np.sin(theta)
y = np.cos(theta)
# 創建一個新的圖形
fig, ax = plt.subplots()
# 繪製  $x^2 + y^2 = 1$ 
ax.plot(x, y, color='#48D1CC', lw=2, label=' $x^2 + y^2 = 1$ ')
# 設定座標軸的比例為1
ax.set_aspect(1)
# 設定y軸單位長度為0.5
ax.set_yticks(np.arange(-1, 1.1, 0.5))
```

十字座標軸程式碼：

```
ax.spines['left'].set_position('zero')
ax.spines['left'].set_color('gray')
ax.spines['left'].set_linewidth(0.5)
ax.spines['bottom'].set_position('zero')
ax.spines['bottom'].set_color('gray')
ax.spines['bottom'].set_linewidth(0.5)
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
```

2. matplotlib 套件

圖 2.14是利用 matplotlib 套件製作的單位圓，python 中的 matplotlib 有專屬套件 Circle 可以直接在座標點上製作任何半徑的圓。

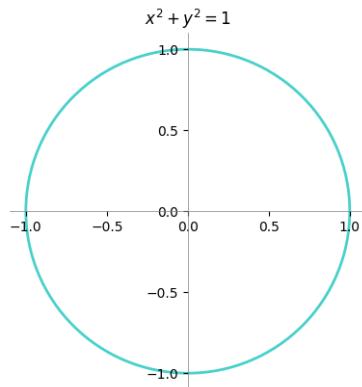


圖 2.14: matplotlib 套件單位圓

利用 Circle 製作圓程式碼：

```
circle = plt.Circle((center_x, center_y),
                     radius, fill=False, color='#48D1CC', lw=2, )
# 將圓形添加到畫布上
ax.add_artist(circle)
```

2.1.11 正方形

1. 繪製四條線法

圖 2.15是分別畫出四條線，限制 x 與 y 的範圍達到畫正方形的效果，是最直觀的畫法，我將四條線分別上不同顏色來辨識。

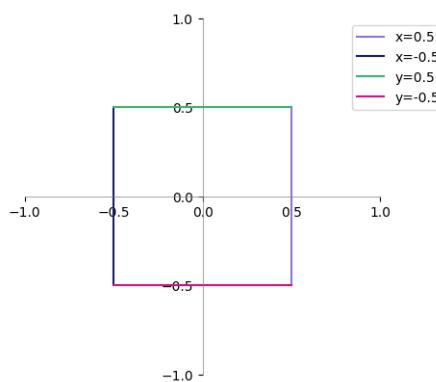


圖 2.15: 四條線法畫正方形

2. 定位四角畫正方形

圖 2.16是先對正方形的四個角的點座標進行定位，再將定位到四角的點連接，來達到畫正方形的效果。

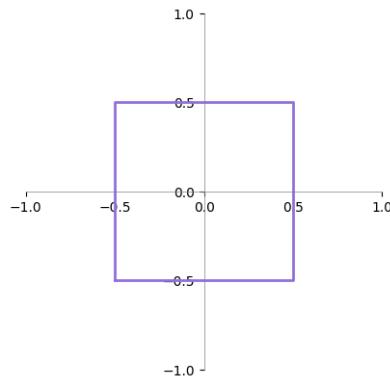


圖 2.16: 定位四角畫正方形

定位四角程式碼：

```
square_points = [(-0.5, -0.5), (-0.5, 0.5),
                  (0.5, 0.5), (0.5, -0.5), (-0.5, -0.5)]
x, y = zip(*square_points)
ax.plot(x, y, color='#9370DB', lw=2)
```

3. matplotlib 套件

圖 2.17是利用 matplotlib 套件製作的正方形，python 中的 matplotlib 有專屬套件 Rectangle 可以在 x 軸座標定位畫製正方形。

```
square = plt.Rectangle((-0.5, -0.5), 1, 1,
                      fill=False, color='#9370DB', lw=2)
# 添加正方形到圖形
ax.add_patch(square)
```

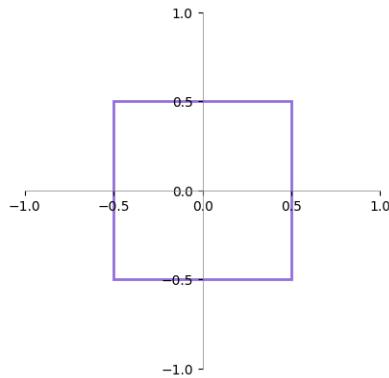


圖 2.17: matplotlib 套件製作正方形

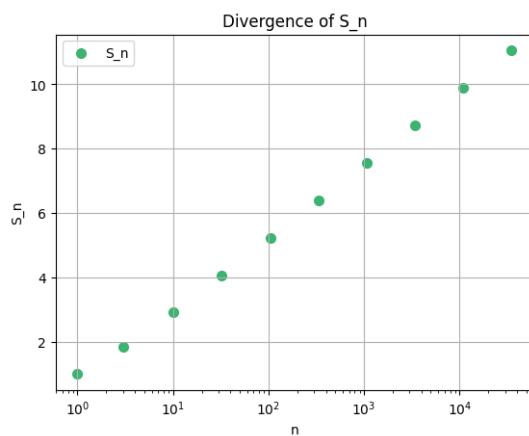
Rectangle 套件用法：

2.2 專題：比較審斂法

令 $S_n = \sum_{k=1}^n \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$

1. 證明 $\lim_{n \rightarrow \infty} S_n$ 收斂

由圖 2.18 得知當 n 值越大 S_n 也會越大，故當 n 趨近無限大時 S_n 會發散。

圖 2.18: $S_n = \sum_{k=1}^n \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$

計算 S_n 並畫圖程式碼：

```
# 計算 S_n
def compute_Sn(n):
    Sn = 0
    for k in range(1, n + 1):
        Sn += 1 / k
    return Sn
n_values = np.arange(1, 10001) # n的範圍，從1到1000
Sn_values = [compute_Sn(n) for n in n_values]
```

2. 設 γ_n 代表陰影區域的總和。證明 $\gamma_n = S_n - \ln(n + 1)$ 。

由圖 2.19 可以得知 γ_n 為紫色線段上的粉紅色區塊相加，而 S_n 為粉紅色區塊相加，計算 $\frac{1}{x}$ 下的面積視為 $\int_1^{n+1} \frac{1}{x} dx$ ，而該積分的結果為 $\ln(n + 1)$ ，因此 $\gamma_n = S_n - \int_1^{n+1} \frac{1}{x} dx = S_n - \ln(n + 1)$ ，故得證。

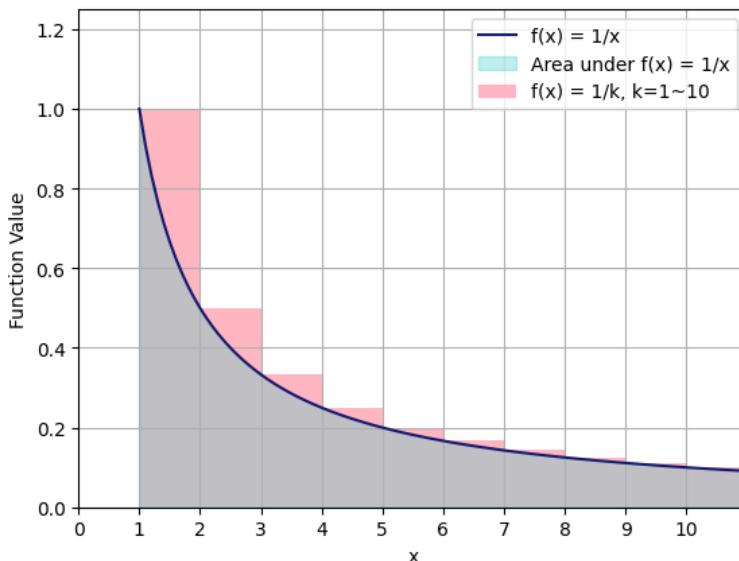


圖 2.19: $\gamma_n = S_n - \ln(n + 1)$

繪製柱狀圖與函數圖形程式碼：

```

# 定義 x 的範圍
x = np.linspace(0.5, 10.5, 100) # 從0.5開始，避免除以0的情況
# 計算 1/x
y = 1 / x
# 計算 f(x) = 1/k，其中 k 的範圍從1到無限大
k_values = np.arange(1, 13) # 調整 k 的範圍，此處設定為1到10
f_x = 1 / k_values
# 繪製 1/x 曲線
plt.plot(x, y, label='f(x) = 1/x', color ='#191970')
plt.bar(k_values, f_x, width=1, label='f(x) = 1/k, k=1~10', color='#FFB6C1', alpha=0.6)

```

3. 證明 $\frac{1}{2}(1 - \frac{1}{n+1}) < \gamma_n < 1$

由圖 2.20明顯可看出如果 n 值是有限的， γ_n 始終夾在 $\frac{1}{2}(1 - \frac{1}{n+1})$ 與 1 之間。

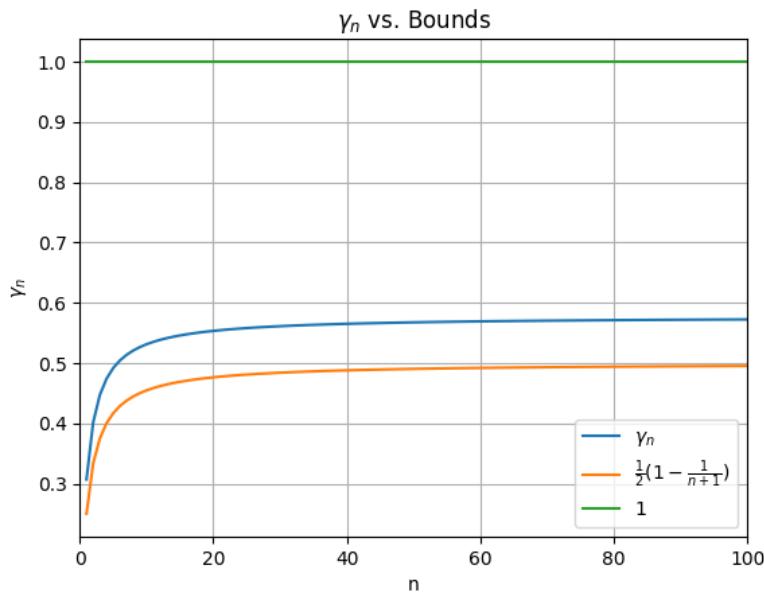


圖 2.20: $\frac{1}{2}(1 - \frac{1}{n+1}) < \gamma_n < 1$

以下用數學證明：

- 右邊不等式 $\gamma_n < 1$:

由圖 2.18 得知 S_n 是發散的，雖然 S_n 的發散速度比 $\ln(n+1)$ 快，但在有限的 n 值條件下， γ_n 是有界的，因此他會小於 1。

- 左不等式 $\frac{1}{2}(1 - \frac{1}{n+1}) < \gamma_n$.⁵

首先我們展開 $\frac{1}{2}(1 - \frac{1}{n+1}) = \frac{1}{2} - \frac{1}{2(n+1)}$ 。我們再將 S_n 用自然對數表示 $S_n \approx \ln(n) + a$ 。我們可以得到 $\gamma_n = S_n - \ln(n+1) \approx \ln(n) + a - \ln(n+1) = a + \ln(\frac{n}{n+1})$, 而 $\gamma_n = a + \ln(\frac{n}{n+1}) > a + \ln(\frac{n}{n}) = a > \frac{1}{2} - \frac{1}{2(n+1)}$ 。

- 根據上述結果得證， $\frac{1}{2}(1 - \frac{1}{n+1}) < \gamma_n < 1$

2.3 結論

Python 中的數學製圖是一個強大的主題，它提供了豐富的工具和函數，讓我們能夠可視化和探索數學的各個方面。透過庫如 NumPy 和 Matplotlib，我們可以輕鬆地生成數學函數、圖形和數據可視化。通過製圖，我們可以更好地理解數學概念，將抽象的數學原理轉化為可視的圖形，使學習變得更加生動和直觀，帶領我們更加深入探索數學之美。這些圖形不僅用於教育，還在科學研究、數據分析和工程應用中發揮著重要作用。總而言之，Python 中的數學製圖為我們帶來了無限的創作和學習機會，同時也促進了數學的應用和理解。

⁵ a 為 Euler's constant 約等於 0.5772

第 3 章

Python 分配圖型多樣性

Python 統計分配多樣性的主題涵蓋了各種機率分配，如常態分配、二項分配、泊松分配等。這個主題重點探討如何使用 Python 程式語言繪製包括數據生成、抽樣、概率質量函數和累積分佈函數的繪製。此外 Python 還提供了各種函數庫和工具，如 NumPy、SciPy 和 Matplotlib，來幫助更有效地進行分配圖型的製作，從而更深入地理解數據多樣性的特性和分佈圖型。

3.1 不同連續與離散分配中的多種可能性

本節我們使用 SciPy 套件，來詮釋包含常態、t、F、柯希... 分配的圖形，來讓我們更加了解分配中的可能性。

3.1.1 常態分配

圖 3.1 將常態分配別分固定 μ 與 σ 來觀察兩者會對圖形有何影響。

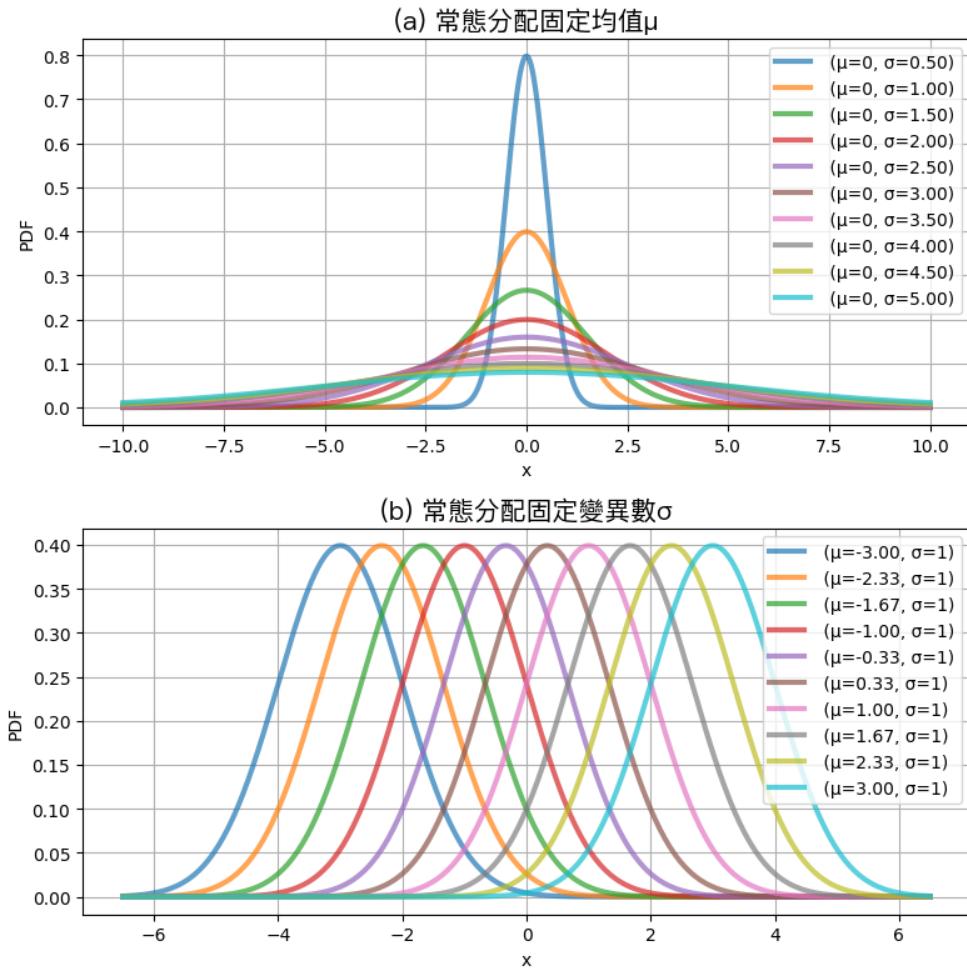
圖 3.1: 常態分配 μ 與 σ 關聯

圖 3.1(a) 可得知當固定 μ 而 σ 增加時，常態的尾巴會越來越長，但頂峰也隨之下降。當固定 σ 而 μ 增加時，常態圖形會平行移動，圖形不變。在統計上時常以常態分配為標準判斷其他分配的圖形特性。

3.1.2 t 分配

圖 3.2 先設定自由度範圍從 0.1 – 30，來觀察 T 分配再不同自由度的情況下會圖形會如何呈現，並且放入常態分配對比。

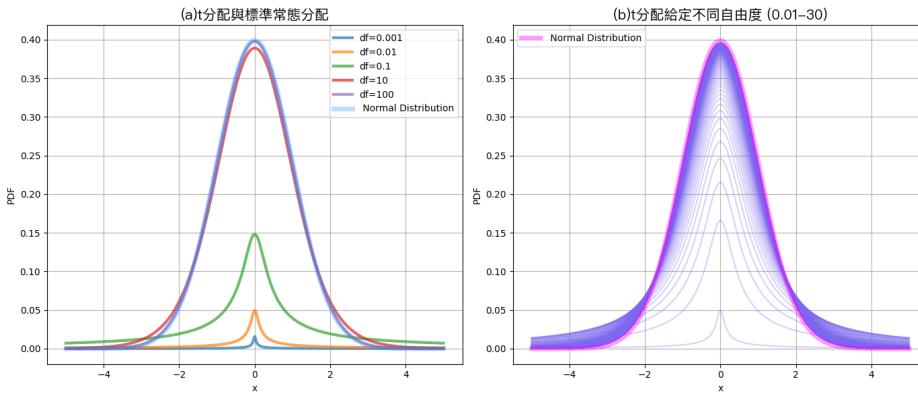


圖 3.2: t 分配與其自由度的關聯

根據圖 3.2(b) 得知 t 分配的特點在於它是圓頭且相較常態分配厚尾，圓頭特性意味著樣本平均值更集中在母體平均值附近。此外，t 分配的尾部相對較長，也就是「厚尾」，這意味著在分佈的尾部有更多的極端值。這種兩種性質使 t 分配適合小樣本統計分析，因為它對異常值或極端值有更強的容忍度。另外從圖 3.2(a) 可知道當自由度越大 t 分配越接近常態。

3.1.3 F 分配

圖 3.3 為 $F(n_1, n_2)$ 兩個自由度互相比較。

- 分子自由度 (n_1): 表示分子變異數的自由度，與分子的組數相關聯。
- 分母自由度 (n_2): 表示分母變異數的自由度，與分母的組數相關聯。

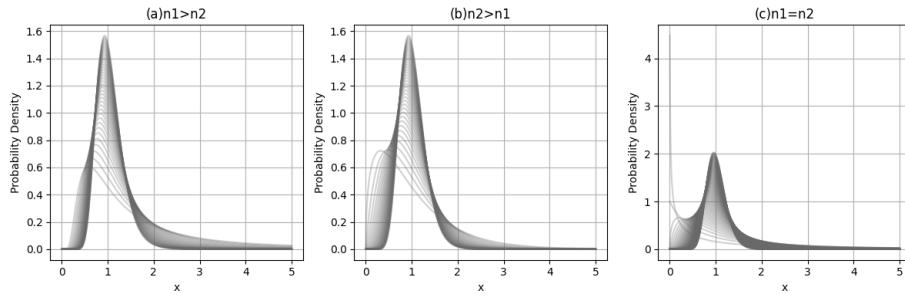


圖 3.3: f 分配自由度

根據圖 3.3(a) 當 $n_1 > n_2$ 時圖形呈現右偏現象厚尾分配，且當 n_1 與 n_2 越接近頂部會變尖。圖 3.3(b) 當 $n_1 < n_2$ 時圖形呈現右偏現象厚尾分配，且當 n_1 與 n_2 越接近頂部也會變尖。圖 3.3(c) 當 $n_1 = n_2$ 時圖形呈現右偏現象厚尾分配，且頂部也是尖頭，且在兩自由度越大時圖型會越接近常態。因此我們可以知道不管兩個自由度如何都不影響特性，F 分配都會是厚尾、右偏且尖頭。F 分配通常用於小樣本的方差分析和回歸分析中的假設檢定，其厚尾特性適合處理方差比較等问题。

3.1.4 柯西分配

圖 3.4 將不同的 α 值的 cauchy 分配與標準常態做比較。可以看出當 α 值越大柯西分配的頂部越高且越尖。而分配的尾巴相對於常態更加的厚重是厚尾的分配，使得柯西分配對機端值非常敏感，因此它在描述極端事件或異常值時很有用。

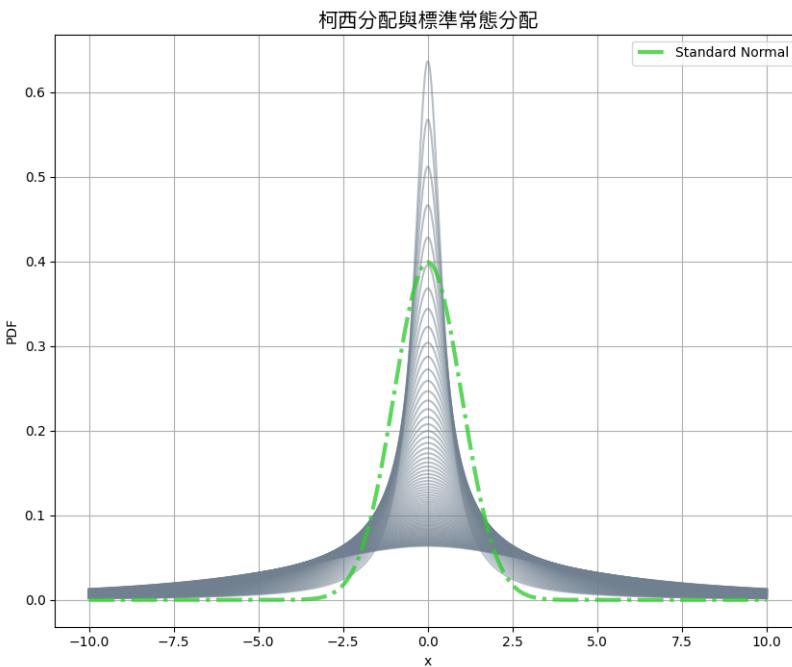


圖 3.4: 柯西分配與常態

上述三個分配都有各自的尾巴與頂部特性，因此我們用表格簡單彙整表 3.1。

表 3.1: 常態、t 和柯西分配比較表

特性	分配			
	常態分配	t 分配	F 分配	柯西分配
尾部	標準	厚尾	厚尾	厚尾
頂部形狀	圓頂	圓頂	尖頂	尖頂

3.1.5 卡方分配

圖 3.5 將不同自由度的卡方分放在一起做比較，並且觀察自由度對圖形趨勢。

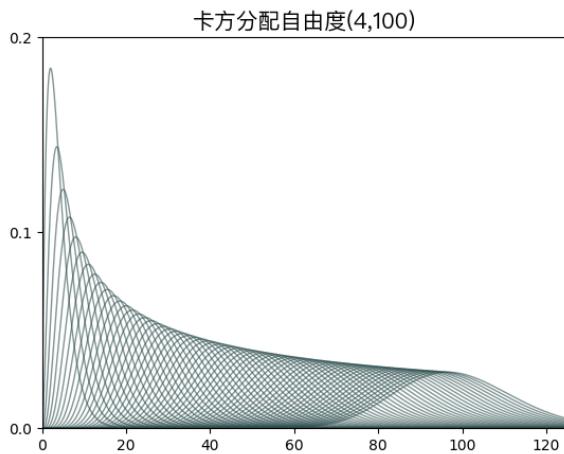


圖 3.5: 多自由度卡方分配圖

根據圖 3.5 可以看出卡方分配的所有值都是正數，因為它通常用於非負數據的變異性。卡方分配形狀由自由度來控制，自由度越大，圖形頂點越往右遞減因此圖形是右偏類型。

3.1.6 貝塔分配

根據圖 3.6 將分配 $\beta(\alpha,\beta)$ 分成固定 α 值改變 β 值，固定 β 值改變 α 值與 $\alpha=\beta$ 三種情況，觀察圖型的特性。由圖 3.6(a) 得知當固定 α 值 β 值越大，大到 $1 \geq \beta > \alpha \geq 0$ 時，圖形呈現右偏分配，且尖峰會出現在接近 0。反之圖 3.6(b) 當固定 β 值 α 值越大，大到 $0 \leq \beta < \alpha \leq 1$ 時，圖形呈現左偏分配，且尖峰會出現在接近 1。圖 3.6(c) $\beta = \alpha$ 分配變得尖峰且對稱，當值不斷增加頂峰也越尖，使其更加集中在中心值。圖 3.6(d) 是將上述三種情況整合在一起。表 3.2 簡單做整理。

表 3.2: Beta 分配比較表

特性	$\beta(\alpha, \beta)$		
	$\beta > \alpha$	$\beta < \alpha$	$\beta = \alpha$
偏向	右	左	無
對稱性	無	無	有
尖峰	0 處	1 處	正中間

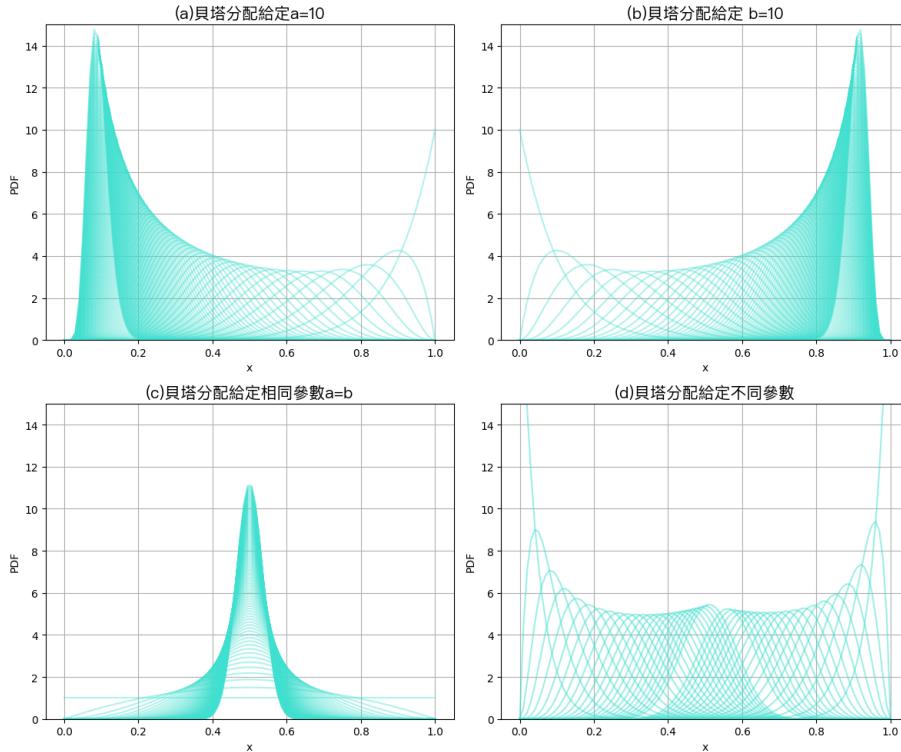


圖 3.6: 不同參數下的 Beta 分配

3.1.7 Gamma 分配

圖 3.7 是一個 $\Gamma(a, b)$ 的圖型，將會分別固定不同得兩參數來做比較，觀察與圖型的關聯。

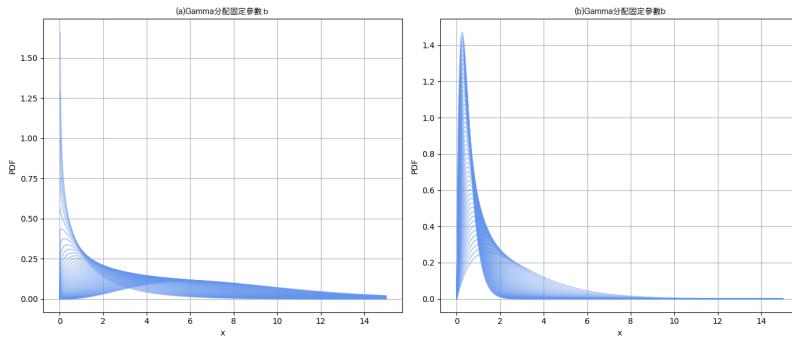


圖 3.7: Gamma 分配與參數關係

- 形狀參數 (a): 決定 Gamma 分配的形狀。當 $a > 1$ 時，分配呈右偏態。當 $a = 1$ ，分配變成指數分配。 $a < 1$ 時，分配呈現左偏態。
- 尺度參數 (b)：決定了 Gamma 分配的尺度。 b 值越大會導致分配的尖峰較高，分配會更加集中在均值附近。反之，分配會更分散

3.1.8 二項分配

圖 3.8 為 $\text{Bin}(10, 0.5)$ 的 PMF 與 CDF，分配的期望值為 $\mu = np$ 變異數 $\sigma^2 = np(1 - p)$ 。由圖 3.8(a) 得知當 $X = 5$ 時為 PMF 的最大值且圖形對稱。再由圖 3.8(b) 因為二項分配為離散，所以 CDF 為階梯函數且最大值為 1。

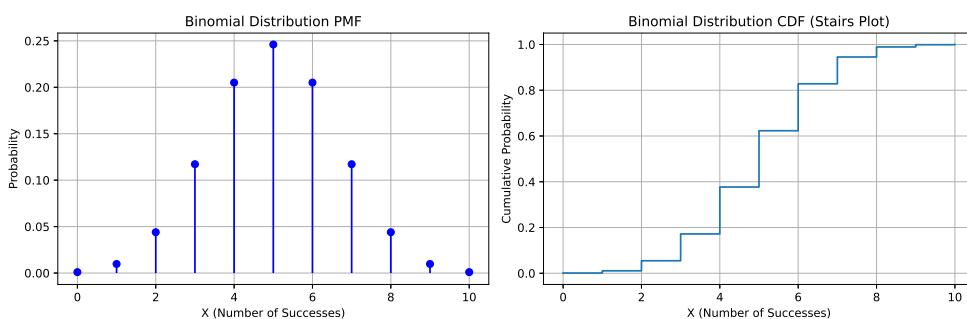


圖 3.8: 二項分配 PMF 與 CDF

3.1.9 柏松分配

圖 3.9 為 $\text{Po}(\lambda = 10)$ 的 PMF 與 CDF，分配的期望值為 $\mu = 10$ 變異數 $\sigma^2 = 10$ 。由圖 3.9(a) 得知當 $X = 9, 10$ 時為 PMF 的最大值我們已柱狀圖來表示。再由圖 3.9(b) 因為柏松分配為離散，所以 CDF 為階梯函數且最大值為 1，利用漸層色來凸顯每層的累積機率並標示出來。

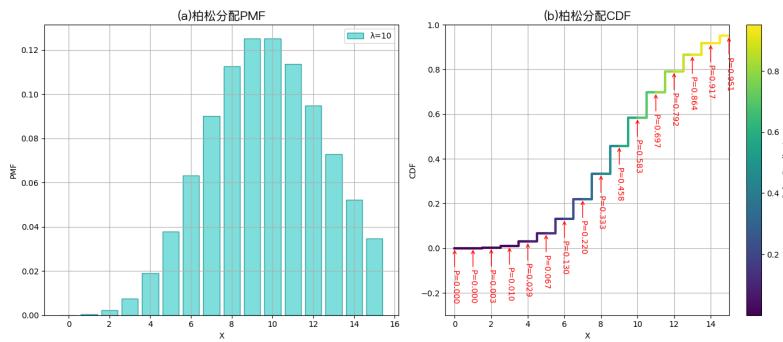


圖 3.9: poissonPMF 與 CDF

3.2 亂數分配分析

利用 Python 選擇不同分配產生多個亂數，在利用不同的分析圖進行分配解析。

3.2.1 亂數生成韋伯分配

圖 3.10 是在亂數下生成韋伯分配的分析為連續型分配，其中參數分別有形狀參數 (γ)=1 與尺度參數 (λ)=2，使用了直方圖、盒線圖、常態機率圖與 CDF 機率圖。

- 直方圖: 根據圖 3.10(a)，可以發現分配的數據大致呈現右偏分配尾部較長，數據更集中在較小的值附近。
- 盒線圖: 根據圖 3.10(b)，可以看到 Weibull 分配的中位數 ≈ 0.7 、分布的分散度與異常值情況。

- 常態機率圖: 根據圖 3.10(c) , 可以發現看到數據點呈現彎曲的趨勢，說明 Weibull 分佈與常態分配不完全匹配。
- CDF 機率圖: 根據圖 3.10(d) , 可以看出 ECDF 曲線的形狀，以及數據分佈的情況。

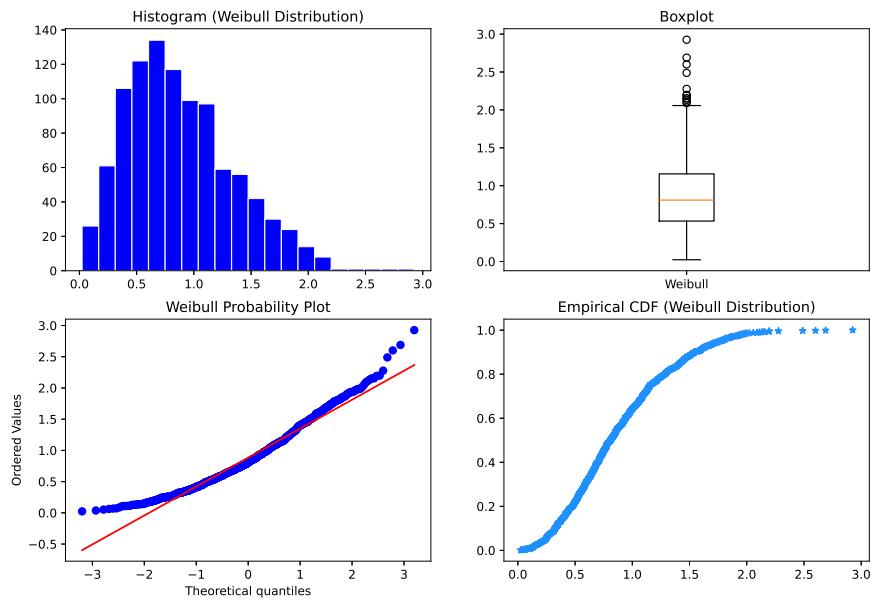


圖 3.10: 亂數生成韋伯分配

3.2.2 亂數生成柏松分配

圖 3.11 是在亂數下生成柏松分配的分析為離散型分配，參數 $\lambda=5$, 使用了直方圖、盒線圖、常態機率圖與 CDF 機率圖。

- 直方圖: 根據圖 3.11(a) , 可以看到柏松分配的數據呈現出一個尖峰，然後迅速減小的形狀，這是柏松分布的典型特徵。
- 盒線圖: 根據圖 3.11(b) , 可以看到數據的中位數接近柏松分布的參數 μ (平均值)，而箱線的長度相對較短，說明數據相對集中在

平均值附近。

- 常態機率圖: 根據圖 3.11(c) , 可以看到數據點不完全沿著直線分佈，這是因為柏松分配與常態分配不完全匹配，但近似線性關係說明在某種程度上相似。
- CDF 機率圖: 根據圖 3.11(d) , 可以看到隨著值的增加，累積機率逐漸增加。圖表可以用来了解數據累積分佈的情況。

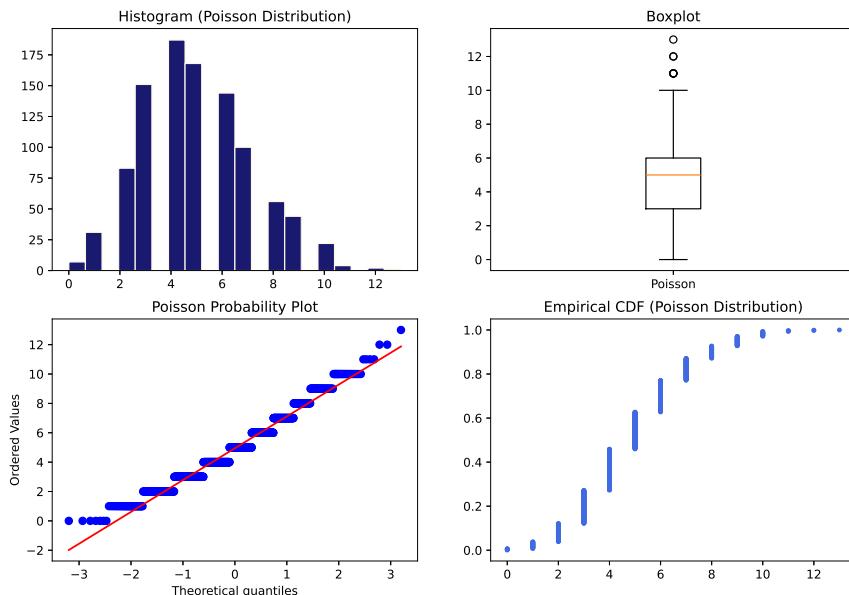


圖 3.11: 亂數柏松

3.3 抽樣分配

抽樣分配是統計學中一個重要的概念，描述了從總體中抽取樣本的統計量的分配情況。有助於我們理解樣本統計量的性質，包括均值、方差等。

3.3.1 中央極限定理

Central Limit Theorem, CLT (中央極限定理)：若隨機變數 $X_i, i = 1, 2, \dots, \infty$ 分布相同且獨立，且其 $(X_i) = \mu, Var(X_i) = \sigma^2 < \infty$, 則

$$\frac{\bar{X}_n - \mu}{\frac{\sigma}{\sqrt{n}}} \xrightarrow{d} N(0, 1)$$

離散分配中的中央極限定理

- 圖 3.12 為離散分配使用的是柏松分配，要證明當樣本數越大樣本平均數越接近真實 pdf。並且更改了柏松的 λ 觀察是否也有關聯。

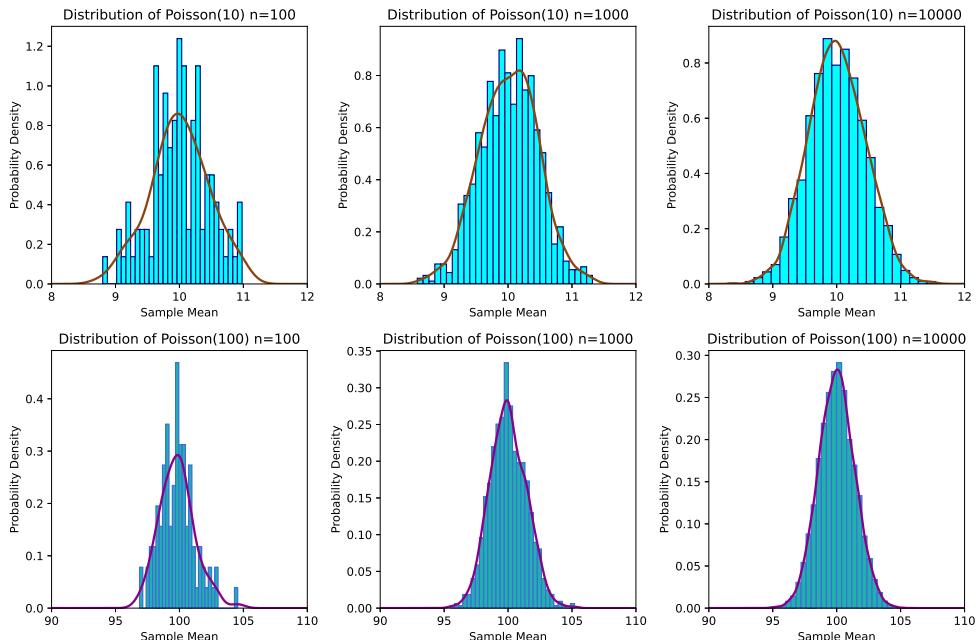


圖 3.12: 離散分配與中央極限定理

根據圖 3.12 分別抽取 $n=100, 1000$ 與 10000 來做對比，並製作出他樣本平均數的柱狀圖，可以明顯發現，當 n 值為 100 時圖外觀參差不齊，當樣本平均數變大圖型越接近真實的 pdf 的線，並且該圖型也趨近於常態。觀察上下圖當柏松 λ 不相同時，其樣本平

均數與真實 pdf 的差異並不大，我們可以用此結果判斷，離散分配在自由度與中央極限定理幾乎互不影響。

2. 由上述可知不管是離散或連續分配，當 n 值越大時圖型就越趨近於常態，而分配中的參數對於趨近的趨勢影響不大。

3.3.2 卡方分配與 F 分配關係

圖 3.13 要觀察兩個卡方如何變或會趨近於 f 分配，並觀察其趨勢。

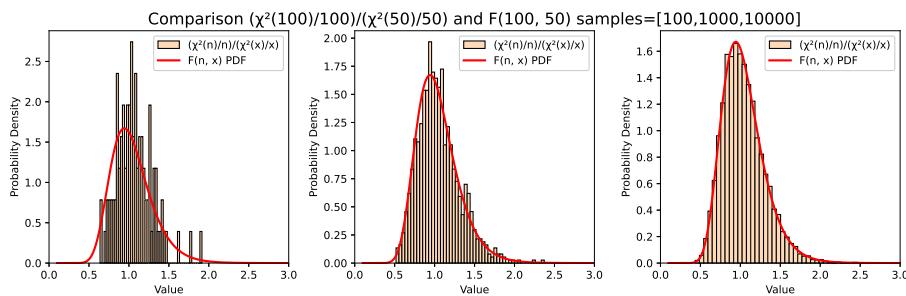


圖 3.13: 卡方分配與 f 分配

根據圖 3.13 當我們抽樣的越多卡方的柱狀圖會越接近 f 分配的 pdf，因此我們可以證實

$$\frac{\frac{x^2(n_1)}{n_1}}{\frac{x^2(n_2)}{n_2}} \approx F(n_1, n_2)$$

◦

3.3.3 Gamma 分配與 Beta 分配

圖 3.14 可以看出 $\frac{\Gamma(a_1)}{\Gamma(a_1) + \Gamma(a_2)}$ 所成的柱狀圖與 $\beta(a_1, a_2)$ 的 pdf 圖，當樣本數越大時兩圖會非常的近似，因此我們可以得知

$$\frac{\Gamma(a_1)}{\Gamma(a_1) + \Gamma(a_2)} \approx \beta(a_1, a_2)$$

◦

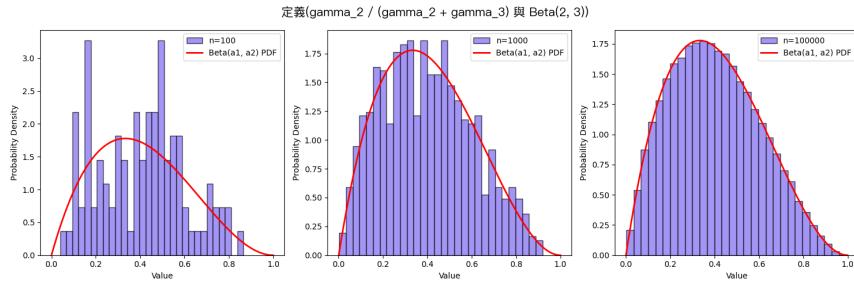


圖 3.14: Gamma 與 Beta 近似關係

3.3.4 二項分配累積性

圖 3.15 亂數產生 100000 抽樣並使用了直方圖、盒線圖、常態機率圖與 CDF 機率圖，來檢視二項的累加性。

- 直方圖: 根據圖 3.15(a) , 顯示了 4 個獨立 Binomial(200, 0.5) 的隨機變量之和的分配與 Binomial(800, 0.5) 進行比較。可看出兩張柱狀圖是非常接近的。
- 盒線圖: 根據圖 3.15(b) , 展示多個二項分配隨機變量之和，可以判斷平均數 $\mu = 400$ 、且圖型呈現對稱，符合二項的特性。
- 常態機率圖: 根據圖 3.15(c) , 可以發現看到數據點幾乎沿著紅線，說明二項分配與常態分配會常近似，因為我們樣本數取值非常大 $n=100000$ ，所以二項理應會趨近於常態。
- CDF 機率圖: 根據圖 3.15(d) , 可以看出 ECDF 曲線的形狀，會非常接近二項的 CDF。

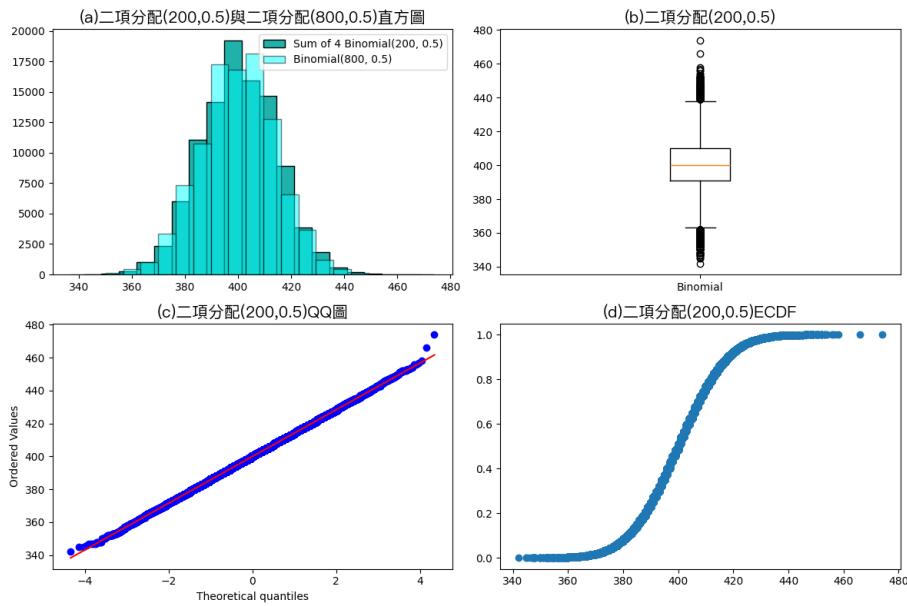


圖 3.15: 二項分配累加性

3.4 四數字隨機抽樣

給定數字 $(2, 4, 9, 12)$ ，從四個數字採抽取放回的方式隨機抽樣。計算其平均數，假設隨機變數 Y 為四數字平均數，並繪製期 PMF。圖 3.17 可以看出圖型呈現對稱的情況，從中間向兩邊遞減，圖形呈現常態狀態。再由圖 3.16 知道最大值在 0.093541 當 $\text{value}=6.75$ 時，也就是四數相加的平均。所以我們可以得知，當抽樣的次數很多接近常態分配。

x 軸為 9.0 時達到最小值: 0.00387
x 軸為 6.75 時達到最大值: 0.093541

圖 3.16: 四數字隨機抽樣最大值

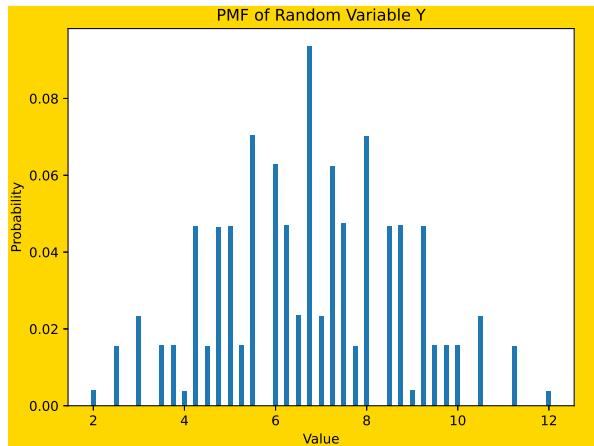


圖 3.17: 四數字隨機抽樣

3.5 結論

Python 中的分配多樣性是統計和機器學習領域中的一個重要主題。我們可以輕鬆地探索和分析不同的分配，例如常態分配、t 分配、柯西分配等。我們可以生成樣本數據，計算統計特性，繪製分佈圖。Python 的豐富庫和工具使我們能夠更深入地了解不同分配的特性，這對於解決現實世界的問題非常有價值。此外，Python 還提供了強大的數據視覺化工具，我們可以將分配多樣性的結果以圖形的方式呈現，使複雜的數據更容易理解和傳達。通過學習和應用 Python 中的分配多樣性相關知識，我們可以更好地理解數據、做出更好的決策，並開發更好的模型來解決各種問題，為我們提供了深入了解和應用數據的能力。

第 4 章

監督式機器學習迴歸模型

機器學習中的迴歸模型是一種監督式學習方法，其模型種類有線性迴歸、加廣型迴歸和羅吉斯迴歸，假設特徵和目標變數之間存在線性關係。本章會討論各種模型在不同條件下的準確率，來判斷回歸模型的好壞，藉以理解分析與資料的關聯。

4.1 機器學習工具

我們使用 PyThon 中的 sklearn 套件的指令，製作的過程中資料可以是現有的或隨機生成的。我們可以更改生成數據，進而判斷在何種資料下哪種迴歸模型準確率最高最適合來分析資料。

sklearn 下載：

```
pip install -U scikit-learn
```

sklearn 套件使用：

```
from sklearn.inspection import  
    DecisionBoundaryDisplay  
from sklearn.linear_model import  
    LogisticRegression  
from sklearn.metrics import confusion_matrix  
from sklearn import metrics
```

4.2 模型介紹

機器學習中的模型指的是通過學習從，數據中模式或進行預測找出該資料的算法或數學結構，分為監督式學習、非監督式學習和強化學習三大類型。本文重點探討監督式學習中的三種迴歸模型。

4.2.1 簡單線性迴歸

線性迴歸是由一個目標因變數 (Y) 與多個自變數 (X) 之間的關係。模型的基本假設是兩個變數之間存在線性關係，目標是找到一條最適合這些樣本點的直線方程式 (4.1)，這是在在雙變數情況下，預測的目標值與實際觀測值之間的差異最小化。

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon \quad (4.1)$$

假設共有 N 筆已知的資料 $([x_1(i)x_2(i)], y(i))$ ，則迴歸係數 $\beta_0, \beta_1, \beta_2$ 以最小平方法求得的最佳解為式 (4.2)

$$\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (4.2)$$

其中

$$\hat{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} X = \begin{bmatrix} 1 & x_1(1) & x_2(1) \\ 1 & x_1(2) & x_2(2) \\ \vdots & \vdots & \vdots \\ 1 & x_1(N) & x_2(N) \end{bmatrix} \mathbf{y} = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix} \quad (4.3)$$

一旦迴歸模型的參數 $\hat{\beta}$ 估計完成，我們說機器完成了學習。便可以拿來對新資料做群組屬性的判別 (預測)。

群組判別：當給予一個新的輸入資料 $x = (x_1, x_2)$ ，根據迴歸模型式 (4.1)，其輸出擬合值為：

$$\hat{y} = \mathbf{X}^T \hat{\beta}, \mathbf{X}^T = [1, x_1, x_2] \quad (4.4)$$

在迴歸模型下的擬合值 \hat{y} 不一定剛好是 0 或 1，因此將 \hat{y} 做分群的動作，最常見的就是以 $\hat{y}= 0.5$ 做為分界。

4.2.2 增廣迴歸

增廣迴歸通常指的是一種擴展簡單線性迴歸模型的方法，其中包括了更多的自變數或者更複雜的模型結構，這個模型可以表示為式 (4.5)

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \beta_4 X_1^2 + \epsilon \quad (4.5)$$

其中 X_1, X_2 是模型的多個自變數， β_0, β_1 是對應的迴歸係數， ϵ 是誤差項。接著計算參數 $\hat{\beta}$ 的最小平方估計，如同線性迴歸模式。在進行增廣線性迴歸分析時，重要的是要理解每個自變數對因變數的影響，以及它們之間的相互關係。此外，模型的擬合和解釋力也是評估模型好壞的重要標準。

4.2.3 邏輯斯迴歸

邏輯斯迴歸是一種用於處理二元分類問題的統計模型。名稱中包含迴歸這個詞，但實際上它是一種分類算法結合回歸概念。邏輯斯迴歸的目標是預測樣本屬於兩個類別中的其中一個的概率，而我們會有其函式如式 (4.6)。

$$\text{Logit}(y) = \ln\left(\frac{y}{1-y}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \quad (4.6)$$

此方法通常沒有最小平方法的代數，所以會以梯度下降法等方法去找出熵最小的地方算出 $\hat{\beta}$ 並以此畫出 $\text{Logit}(\hat{y} = 0.5) = \mathbf{X}\hat{\beta}$ 做為分界線。

4.3 回歸模型特性與優劣

本節將要探討，給定兩個二維常態分配來生成資料，並利用平均數、變異數與樣本數不同的條件下模型分類的準確率，且比較不同學習器的好壞。

4.3.1 匯入資料與簡單說明

圖 4.1 為資料 la_3 來做學習器的比較，當中 $y = 0$ 為藍點 $y = 1$ 為紅點，圖中藍色實線是由線性迴歸模型畫出的分割線，準確的機率為 0.7300，而綠色虛線則是由增廣型迴歸模型所繪製的等高線，其準確的機率為 0.7250，粉色虛線則是羅吉斯回歸模型的分割線，其準確的機率為 0.7300，由此可以看出該資料，線性與邏輯斯是比增廣來得更準確一點。

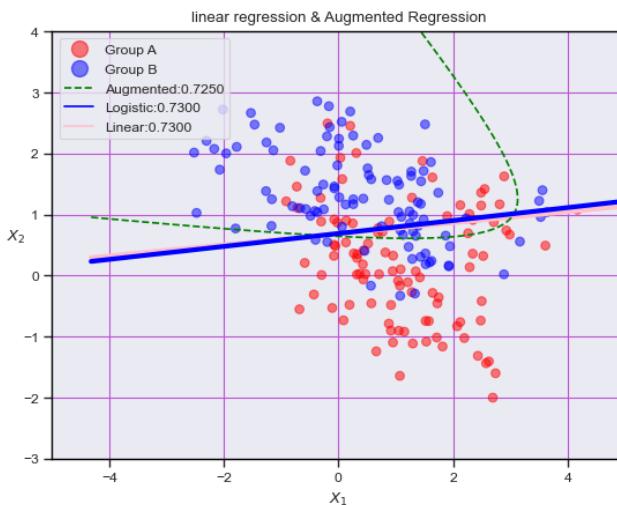


圖 4.1: 匯入資料做迴歸

製作線性迴歸模型順序：

1. 建立模型：創建一個線性迴歸模型，使用 scikit-learn 中的 LinearRegression。

2. 模型訓練：訓練數據 X 和目標變數 y 進行模型訓練，找到最佳的參數，包括權重係數 b 和截距項。
3. 生成邊界：利用模型參數決策邊界直線，這條直線將特徵空間分為兩個區域。
4. 模型預測：訓練好的模型對訓練數據 X 進行預測，得到估計值 \hat{y} 。
5. 概率轉換：將預測轉換為二元類別，通常是根據 0.5 的值，大於 0.5 的部分被標記為 1，小於等於 0.5 的部分被標記為 0，形成預測結果 y_{pre} 。

製作增廣迴歸模型僅與線性模型基本相同請不同於兩處：

1. 配適的模型會不一樣，因為多了 X_1X_2, X_1^2, X_2^2 的項數。
2. 增廣模型通常用於處理非線性分類問題，其中特徵的組合（如交互項和高次項），這種模型在二維空間中可能難以直觀呈現，因為其決策邊界可能是曲線或曲面，故利用等高線模型來呈現較佳。

4.3.2 兩群組之距離近

圖 4.2 為各 200 筆的兩個二維常態分配的隨機變數，當群組的平均值相近，其餘條件皆不變的情況下。

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

由圖 4.2 可看出這個模型訓練的準確度都不高，皆在六成左右，表示當平均數太相近時，會導致資料混雜在一起，而準確率就會下降。

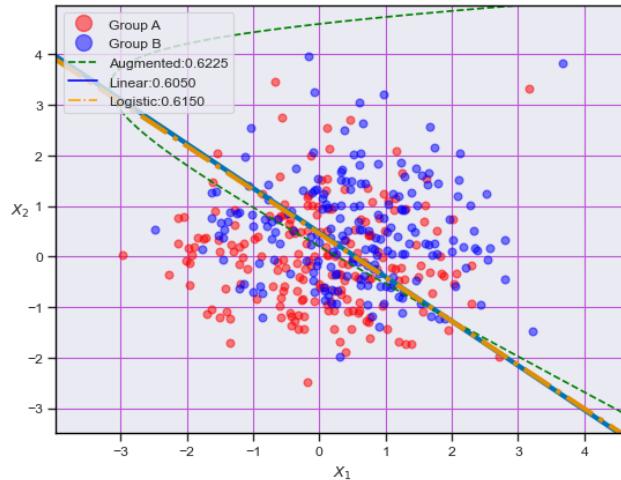


圖 4.2: 群組平均差距小

4.3.3 兩群組之距離遠

圖 4.3 為各 200 筆的兩個二維常態分配的隨機變數，當群組的平均值相對圖 4.2 更加分開，看其準確率會是如何。

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

由圖 4.3 可以看出相比之前他模型的準確率比較好了線性達到 0.9250 而增廣達到了 0.150，邏輯斯準確率也有到 0.92。三種學習器都會分常優良，可以看出當平均數差距變大時，資料就容易往兩側靠攏，而準確度就跟著上升。根據上述也可以推測，當平均數差距夠大時，圖型會有兩極化的現象，使準確率達到 100%，但這樣就失去比較的意義了。

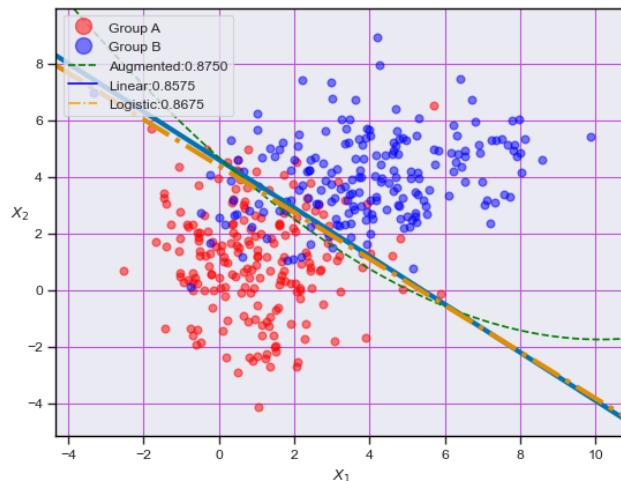


圖 4.3: 群組平均差距大

4.3.4 共變數矩陣較大

圖 4.4 為各 200 筆的兩個二維常態分配的隨機變數，當資料的共變異數矩陣比較大時，觀察準確率會是如何。

$$(x_1, x_2) \sim \boldsymbol{\mu} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \Sigma = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$

$$(x_1, x_2) \sim \boldsymbol{\mu} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \Sigma = \begin{pmatrix} 3 & 1 \\ 1 & 4 \end{pmatrix}$$

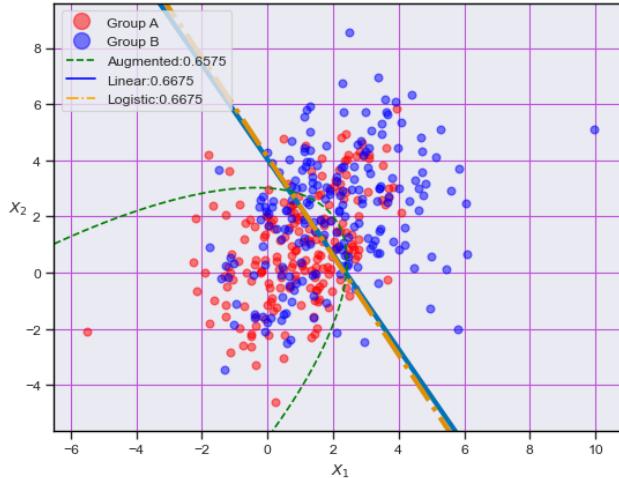


圖 4.4: 群組變異數矩陣值大

由圖 4.4 可以看出因為資料點較為分散，所以導致資料重疊嚴重使三個學習器的準確率都偏差。

4.3.5 共變異數矩陣較小

圖 4.4 為各 200 筆的兩個二維常態分配的隨機變數，當資料的共變異數矩陣相對 4.4 比較小時，觀察準確率變化。

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \Sigma = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.1 \end{pmatrix}$$

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \Sigma = \begin{pmatrix} 0.7 & 0.5 \\ 0.5 & 0.6 \end{pmatrix}$$

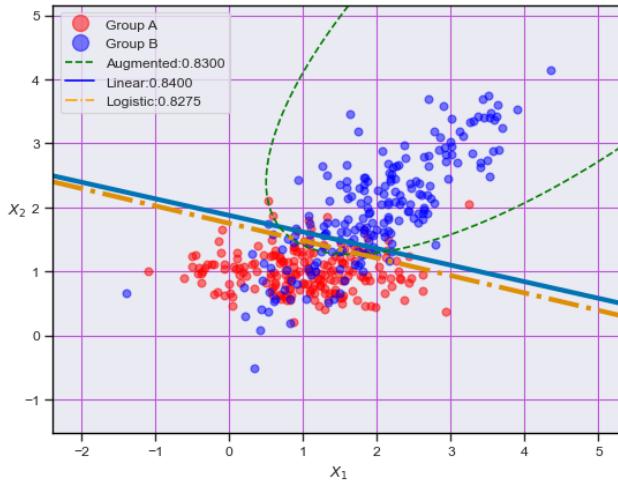


圖 4.5: 群組變異數矩陣值小

由圖 4.5 課以看出同一組平均數下，因為共變異數矩陣較小的原因，資料點會集中在平均數附近，使得準確率都會很好。增廣模型準確率 0.83、線性模型 0.84 和邏輯斯模型也有 0.827，對此組資料來說線性模型比較佔優。

4.3.6 樣本數

由圖 4.6 將平均數與變異數固定下，改動不同樣本數來觀察。

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}, n = 100$$

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}, n = 200, 500, 1000$$

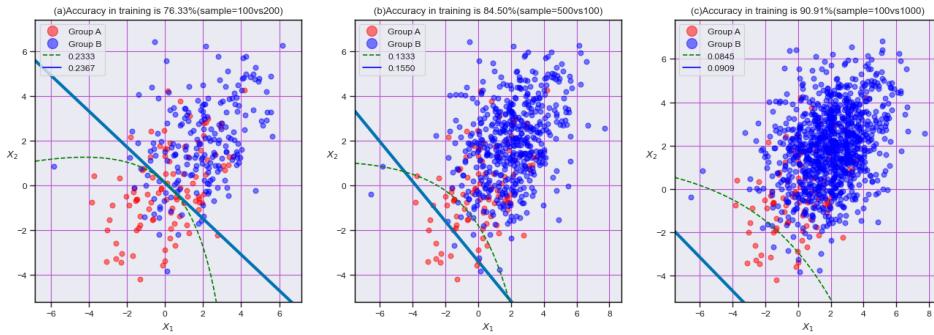


圖 4.6: 樣本數 100vs200,500,1000 比較圖

圖 4.6 可以看出如果平均與變異數值都良好的情況下，更動樣本數量對於準確率影響不大，但樣本越多準確率也會跟著有所提升。雖然線性與增廣在這種條件下錯誤率都非常低，但還是可以比較出增廣模型比線性來得更加優秀。較大的樣本數提供了更多的資訊給模型，有更多的觀察數據以學習模式，從而更好地捕捉數據中的結構和變異性，但就改動樣本數一個變數我們可以說其實對準確率影響力不大。

4.3.7 樣本相依

圖 4.7 將兩組資料數據設定為相依的情況下，觀察圖型會如何變動。

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & 0.2 \\ 0.2 & 1 \end{pmatrix}$$

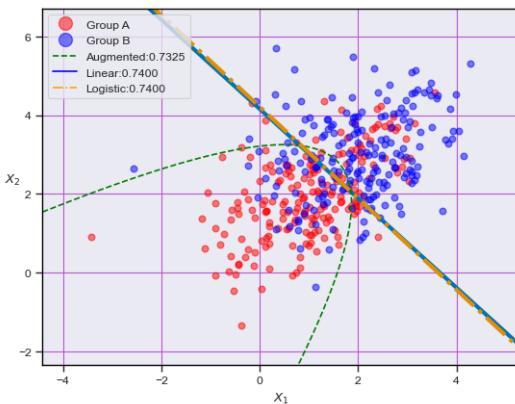


圖 4.7: 資料共變數矩陣相依

由圖 4.7 可以看出相依資料的準確度也是一般，在線性與增廣模型兩者比較下，增廣是較佳的。相依性可以提供額外的信息，使得模型更容易捕捉到數據中的模式，模型可能能夠更好地預測一個變數基於其他變數的變化。這種情況下，相依性有助於模型更好地適應數據。另一方面，如果相依性過強，模型可能會受到多重共線性的影響，這可能使得模型有不穩定的風險。

4.3.8 比較

較優模型	數據變數			
	平均數遠	平均數近	變異數值大	變異數值小
線性	✓	都不好	都不好	
增廣	✓	都不好	都不好	✓

較優模型	數據變數		
	變異數一大一小	樣本數	相依
線性		不影響	
增廣	✓	不影響	✓

4.3.9 兩類資料邏輯斯迴歸模型

圖 4.8 為生成兩組數據，來做邏輯斯回歸。可以知道此組資料有高的準確率，且邏輯斯模型的錯誤率極低，是個好的模型假設。

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 4 \\ 1 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & 0.2 \\ 0.2 & 1 \end{pmatrix}$$

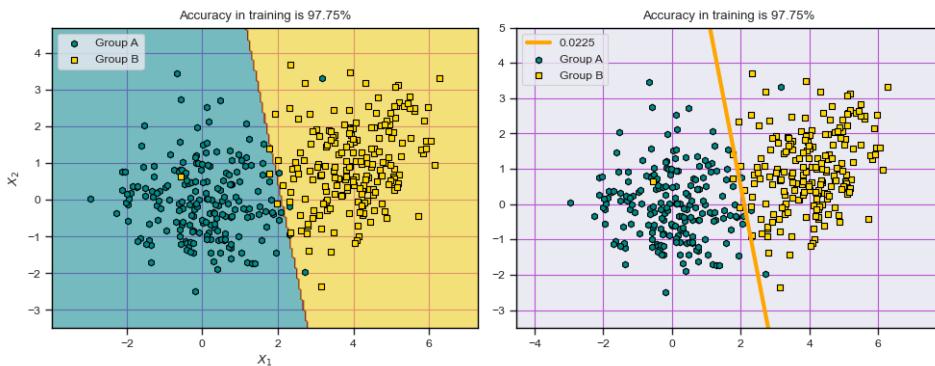


圖 4.8: 兩組資料邏輯斯迴歸模型比較圖

生成邏輯斯回歸模型流程：

1. 模型生成：創建邏輯斯模型，將資料 X 和目標變數 y 用 fit 方法進行模型擬合。
2. 決策邊界：模型中獲取係數與截距項，並使用參數計算決策邊界。
3. 模型預測：模型對資料 X 進行預測，根據預測值計算二元分類的預測結果，將預測概率大於 0.5 的歸為類別 1，否歸為類別 0。

4.3.10 三類資料邏輯斯迴歸模型

圖 4.9 為三組資料生成不同樣本數，做邏輯斯迴歸，來觀察他的準確率，並以顏色來劃分各自的區塊。

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 10 & 0 \\ 0 & 1 \end{pmatrix}$$

$$(x_1, x_2) \sim \mu = \begin{pmatrix} 5 \\ 3 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 2 & 0.9 \\ 0.9 & 3 \end{pmatrix}$$

$$(x_1, x_2) \sim \mu = \begin{pmatrix} -1 \\ 5 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & -0.9 \\ -0.9 & 2 \end{pmatrix}$$

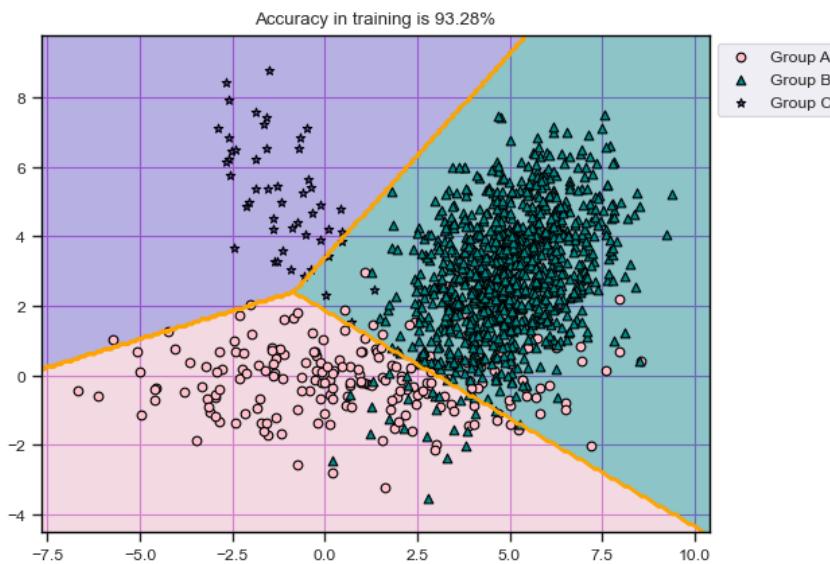


圖 4.9: 三組資料邏輯斯

由圖 4.9 看出藉由邏輯斯模型來對資料做分群，其準確率達 93.28%。可能的是因為這三筆資料，其平均數本身就有差異，故使用模型時準確率必然不會太低。

4.4 訓練資料與測試

圖?? 是將資料 la_1 分成 8:2 的數量來做訓練與測試，來看看同比資料下，不同模型預測是否準確。由圖?? 我們在視覺上圖型大致上是相同，在觀查個別迴歸模型錯誤率方面，雖然有變動但都不大，表示訓練與測試分別跑出的結果相近，也代表訓練是成功的。

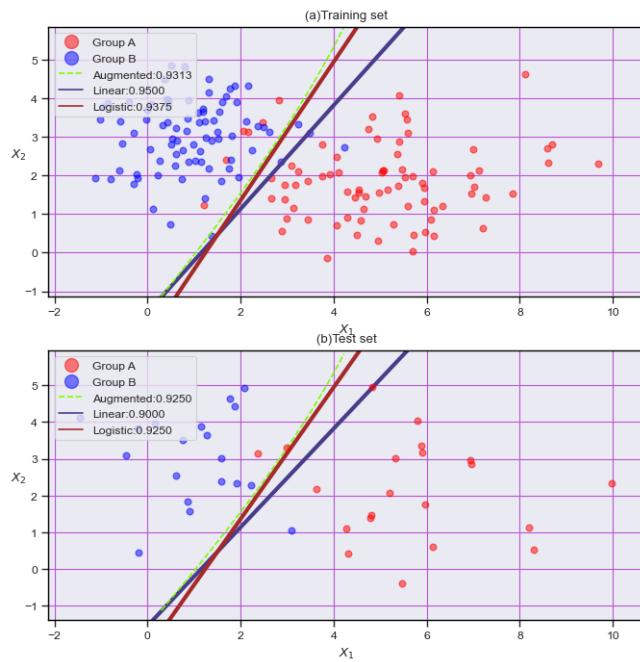


圖 4.10: 訓練與測試

4.5 結論

迴歸模型是一種重要的預測工具，可方便探索變數之間的關係並預測目標值，不同的迴歸模型適合處理不同的資料類別。若數據呈現線性趨勢，線性迴歸模型可能是一個簡單選擇；若數據複雜度較高，可以考慮使用增廣迴歸模型以更好找出。而邏輯斯迴歸模型則適合處理分類問題，預測樣本屬於不同類別的機率。透過深入理解數據分佈和模型性質，我們能更精確地選用合適的迴歸模型以提高預測性能。

第 5 章

監督式學習器與應用

本章討論三個學習器，線性判別分析（LDA）、二次判別分析（QDA）和 K 鄰近模型（K Nearest Neighbors）是常用的分類演算法，皆與後驗機率概念結合。LDA 和 QDA 是統計模型的方法，兩者主要差異在於對共變數矩陣的設定，兩者都尋求找到最佳的線性或二次判別。相比之下，KNN 則是一種基於鄰近度的非參數演算法，通過比較待分類點與鄰近訓練資料的距離，由最近的 K 個鄰近點決定類別，可取不同的 K 值做測試。這三種方法在處理不同資料特性和分佈時表現不同，選擇適當的演算法取決於資料性質和需求。

5.1 線性判別分析 (LDA)

¹假設 X 為多變量資料樣本變數， G 代表群組的類別變數，則後驗機率寫為 $P(G = k|X = x)$ 。在計算資料屬於不同群組的機率後，得到最大機率的群組便是該資料屬於的組別，如式 (5.1) 也稱為判別式分析 (Discriminant Analysis)。

$$G(x) = \operatorname{argmax}_k \ln Pr(G = k|X = x) \quad (5.1)$$

¹本文引用：<https://ntpuccw.blog/python-in-learning/sml-lesson-2-linear-and-quadratic-discriminant-analysis-lda-qda-probabilistic-approach/>

因為式 (5.1) 太過直覺且難以計算應用，故可用貝氏機率中的後驗機率來改寫成如式 (5.2)。

$$P(G = k|X) = \frac{P(X|G = k)P(G = k)}{\sum_l P(X|G = l)P(G = l)} \quad (5.2)$$

其中 $P(X|G = k)$ 表示第 k 組資料發生的機率密度函數，而 $P(G = k)$ 代表群組 k 發生的機率。相較於不知從何計算的後驗機率 $P(G = k|X)$ ，這兩個機率函數比較容易從已知的資料中估計得到。本章假設群組的機率密度函數 $P(X|G = k) = f_k(X)$ 服從多變量常態分配 (Multivariate Normal Distribution)，如式 (5.3)。

$$P(X|G = k) = f_k(X) = \frac{1}{(2\pi)^{\frac{p}{2}} |\sum_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(X - \mu_k)^T \sum_k^{-1} (X - \mu_k)} \quad (5.3)$$

資料變數 $X \in R_p$ ， μ_k 與 \sum_k 分別代表第 k 群資料常態假設的均值與共變異矩陣。簡化問題的複雜性，假設所有群組的共變異矩陣都相等，即 $\sum_k = \sum, \forall k$ 。加入貝氏定理與資料常態假設後，式 (5.1) 的最大後驗機率問題改寫為式 (5.4)。

$$\begin{aligned} G(x) &= \arg \max_k \ln Pr(G = k|X = x) \quad (\text{by Bayes' theorem}) \\ &= \arg \max_k \ln(Pr(X = x|G = k)Pr(G = k)) \quad \text{by (5.3)} \\ &= \arg \max_k \ln\left(\frac{1}{(2\pi)^{\frac{p}{2}} |\sum_k|^{\frac{1}{2}}}\right) - \frac{1}{2}(X - \mu_k)^T \sum_k^{-1} (X - \mu_k) \\ &\quad + (\ln(Pr(G = k))) \\ &= \arg \max_k x^T \sum^{-1} \mu_k - \frac{1}{2} \mu_k^T \sum^{-1} \mu_k + \ln Pr(G = k) \end{aligned} \quad (5.4)$$

其中式 (5.4) 因為 $\ln\left(\frac{1}{(2\pi)^{\frac{p}{2}} |\sum_k|^{\frac{1}{2}}}\right)$ 與 k 無關視為常數，不影響最大值，因此可以省

略，最終得到的目標函數也稱作線性判別式函數如式 (5.5)。

$$\sigma(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \ln Pr(G = k) \quad (5.5)$$

式 (5.5) 中，只有 X 已知，其餘未知，但可以由已知去推估未知，其估計如下。

1. $Pr(G = k) \sim \hat{\pi}_k = N_k/N$, N_k 代表第 k 組資料總數， N 代表所有的總數量。
2. $\hat{\mu}_k = \sum_{G=k} x_i / N_k$ 。
3. $\hat{\Sigma} = \sum_{k=1}^K (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T / (N - K)$ ， K 為群組個數， N 為所有群組的資料總數，此估計也稱作 Pooled within-group covariance matrix。

回歸本章節主題，繪製 LDA 分界線，在概念上我們喜歡從二維資料去繪製平面或空間的群組分界線，再推至 p 度空間的群組間的分界線。因此必須先找出兩個群組的共同條件，譬如，群組 k 與 l 分界線滿足式 (5.6)。

$$Pr(G = k | X = x) = Pr(G = l | X = x) \quad (5.6)$$

在後驗機率相等的群組分界原則下，經過移項取 \ln ，結合由貝式定理與資料的常態分配假設可得分界線的式子如 (5.7)。

$$\ln \frac{Pr(G = k | X = x)}{Pr(G = l | X = x)} = \ln \frac{f_k(x)}{f_l(x)} + \ln \frac{P(G = k)}{P(G = l)} \quad by(5.2)$$

$$= \ln \frac{P(G = k)}{P(G = l)} - \frac{1}{2} (\mu_k + \mu_l)^T \Sigma^{-1} (\mu_k - \mu_l) \\ + x^T \sum_{i=1}^{n-1} (\mu_k - \mu_l) \quad by(5.3)$$

$$= 0 \quad (5.7)$$

於是從資料的後驗機率相等，得到式 (5.7) 的線性方程式，決定群組的

分野，在資料為二度空間的維度裡，這條線性的分界線可以被畫出來如圖 5.1。

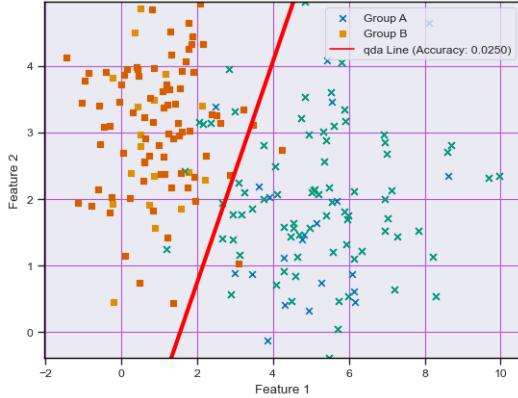


圖 5.1: 資料 la_1 LDA 分界線

5.2 二次判別分析 (QDA)

如前段所述，LDA 與 QDA 差異來自共變異矩陣相同的假設。當令各群的共變異矩陣不同時可以推導出式 (5.8)。

$$\begin{aligned}
 G(x) &= \arg \max_k \ln Pr(G = k | X = x) && \text{(by Bayes' theorem)} \\
 &= \arg \max_k \ln(Pr(X = x | G = k) Pr(G = k)) && \text{by (5.3)} \\
 &= \arg \max_k \ln f_k(X) + \ln Pr(G = k) - \ln \sigma_l f_l(X) Pr(G = l) \\
 &= \arg \max_k -\frac{1}{2} \ln |\sigma_k| - \frac{1}{2} (X - \mu_k)^T \sigma_k^{-1} (X - \mu_k) \\
 &\quad + \ln Pr(G = k) - \ln \sigma_l f_l(X) Pr(G = l) \\
 &= \arg \max_k -\frac{1}{2} \ln |\sigma_k| - \frac{1}{2} (x - \mu_k)^T \sigma^{-1} (x - \mu_k) + \ln Pr(G = k)
 \end{aligned} \tag{5.8}$$

其中因為 $-\ln \sigma_l f_l(X) Pr(G = l)$ 為一個減掉得正數值，在最大值中可以去掉不考慮，則如式 (5.5) 的線性判別式函數將改寫為式 (5.9)。

$$\delta_k(x) = -\frac{1}{2} \ln |\Sigma_k| - \frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) + \ln \Pr(G = k) \tag{5.9}$$

式 (5.9) 稱作稱作二次判別式函數, 因群組 k 與群組 l 之間的分界線不是直線, 而是變數的二次方所以稱為二次, 此分界線寫成式 (5.10)。

$$\{x|\delta_k(x) = \delta_l(x)\} \quad (5.10)$$

令 $x = [X_1 X_2]$ 時, 可將式 (5.10) 推導成式 (5.11), 為 $x = [X_1 X_2]$ 平面上的二次曲線如圖 5.2。

$$c = c_1 X_1 + c_2 X_2 + c_3 X_1 X_2 + c_4 X_1^2 + c_5 X_2^2 \quad (5.11)$$

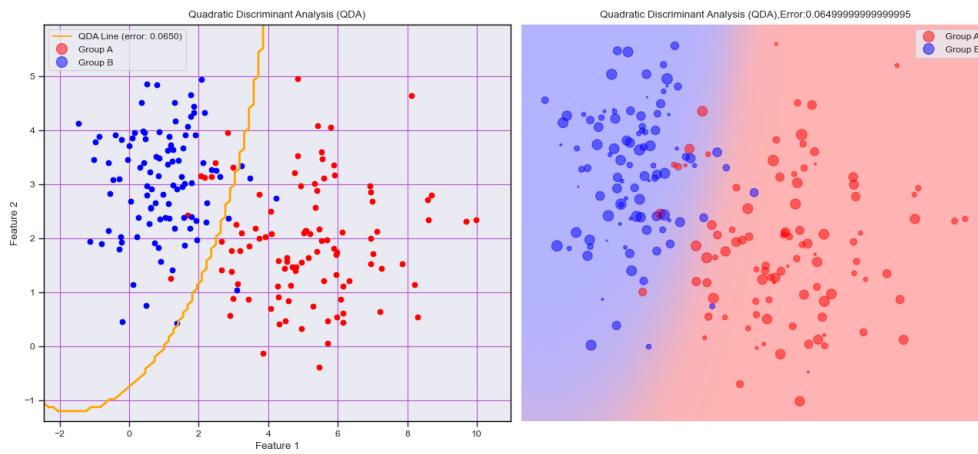


圖 5.2: 資料 la_1 QDA 分界線

該範例與圖 5.1 不同我們使用顏色來區分出 QDA 的分界線, 明顯看出 QDA 是曲線來分界的。

5.3 K 鄰近模型 (KNN)

設 Y 和 $f(X)$ 是輸出變數和輸出預測值, 其中 $X \in \mathbb{R}^p$ 表示有 p 個輸入變數。期望輸出值與預測值的差距越小越好, 若輸入變數 X 與輸出變數 Y 的聯合機率密度函數 $P(XY)$ 已知, 可以把上述狀況轉換成式 (5.12)

◦

$$\min_{f(x)} E_{XY}[(Y - f(X))^2] \quad (5.12)$$

若給定 $X = x$ 可推導出式 (5.13)。

$$\hat{f}(x) = E_{Y|X}(Y|X = x) \quad (5.13)$$

但計算出式 (5.13) 過於困難，因此使用到平均值來估計此期望值，得到式 (5.14)，稱作 K 鄰近模型，從已知的資料中找到 K 個最靠近 x 的資料，將這些鄰近的 K 筆資料所對應的 y 值平均起來作為「條件式均值」的估計。

$$\hat{y} = \text{Ave}(y_i | x_i \in N_k(x)) = \frac{1}{K} \sum_{x_i \in N_k(x)} y_i \quad (5.14)$$

由於是類別資料的關係，其輸出群組變數改寫為 G ，預測群組寫成 $g(X)$ ，兩者的誤差以「Loss function」 $L(G, g(X))$ 取代原先的平方差。

$$L(G, g(X)) = \begin{cases} 0 & \text{if } G = g(x) \\ 1 & \text{if } G \neq g(x) \end{cases} \quad (5.15)$$

因此 (5.14) 的最佳解為式 (5.16)：

$$\hat{g}(X) = g_k \text{ if } Pr(g_k | X = x) = \max_{g \in G} Pr(g | X = x) \quad (5.16)$$

當假設兩個群組的輸出 g_1 為 0 與 g_2 為 1 時，式 (5.14) 可以當作式 (5.16) 的估計式，並配合下列的群組判別式 (5.17) 畫出分界線如圖 5.3。

$$x \in \begin{cases} g_1 & \text{if } \hat{y} \leq 0.5 \\ g_2 & \text{if } \hat{y} > 0.5 \end{cases} \quad (5.17)$$

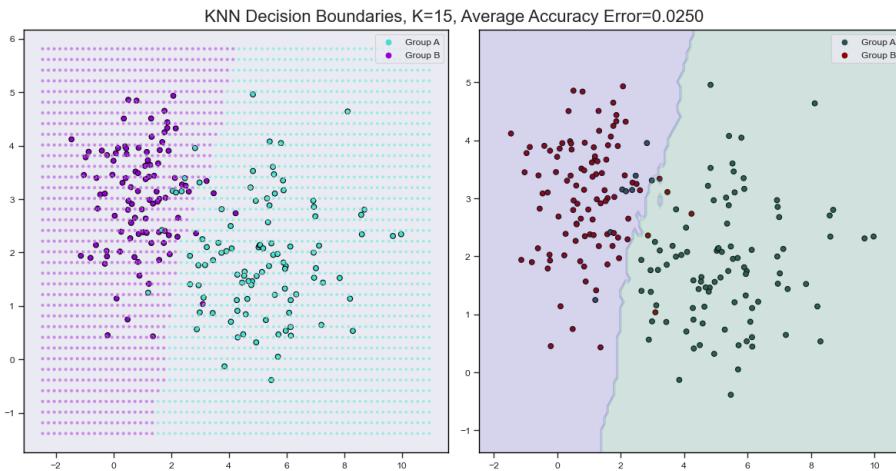


圖 5.3: 資料 la_1 ,K=15 時 KNN 分析

5.4 兩群資料之學習器評比

生成的資料來自多變量常態母體，本章節會著重在共變數矩陣，對學習器有何影響，進行不同的假設來進行評比。並將訓練資料與測試資料分為 8:2 且進行 Bootstrapping 100 次計算平均誤判率。

5.4.1 兩群共變數矩陣相同

狀況一生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 2 \\ 5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

由圖 5.4 與表 5.1，可以看出在變異矩陣相同的情況下 LDA 三項數據都好過其他三種學習器，也可以證明當共變數矩陣相同時 LDA 是相當好用的選擇。而再不同的 K 值下，當 K=15 時訓練的錯誤率是比 K=20 來的更加優秀的，但其餘兩個值都相同，原因在於資料比較兩極化平均距離較大，分類明顯。

表 5.1: 相同共變數矩陣在不同學習器下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=15)	KNN(K=20)
原資料錯誤率	0.0775	0.0825	0.0875	0.0825
訓練錯誤率	0.08125	0.0875	0.0843	0.0855
預測錯誤率	0.0625	0.0683	0.0749	0.0749

5.4.2 兩群資料共變數不同且正相關

狀況二生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 4 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

由圖 5.5 與表 5.2，可以看出在不同共變數矩陣下 QDA 在三個數據方面是好過 LDA，也間接證明如果共變數矩陣沒有相同，LDA 會遜色於 QDA。而原資料與訓練錯誤率 K=15 都優於 K=20，但在預測的錯誤率上兩者是一樣的，著重在看預測的錯誤率因此我們可以說模型的配飾剛好對於 K=15 是好的，但預測能力上兩者相同。

表 5.2: 不同共變數矩陣且正相關在不同學習器下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=15)	KNN(K=20)
原資料錯誤率	0.1850	0.1575	0.1500	0.1525
訓練錯誤率	0.1968	0.1531	0.1343	0.1406
預測錯誤率	0.1650	0.1500	0.1624	0.1624

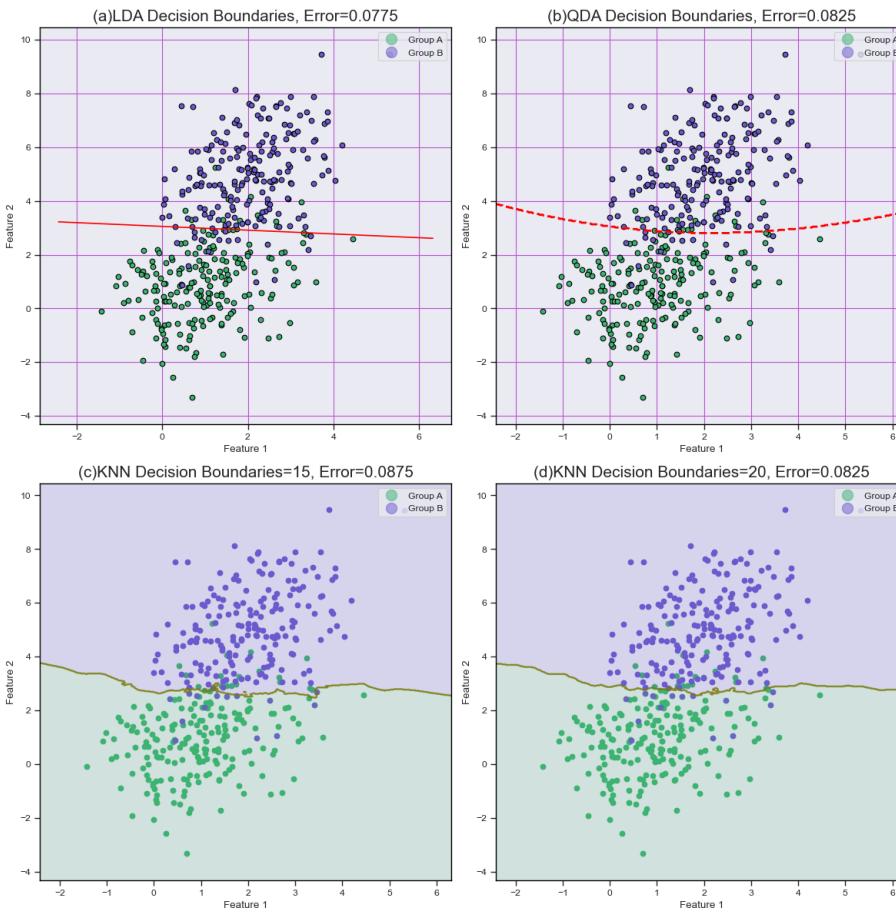


圖 5.4: 共變數矩陣相同下 LDA,QDA,KNN=15 與 KNN=20 比較圖

5.4.3 兩群資料共變數不同且相關性一正一負

狀況三生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 3 & -0.5 \\ -0.5 & 2 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 4 & 0.2 \\ 0.2 & 2 \end{pmatrix}$$

由圖 5.6 與表 5.3，可以看出在不同共變數矩陣下不管正相關或負相關 QDA 都會優於 LDA。由原資料與訓練來觀察，當 K=15 與 K=20 說是

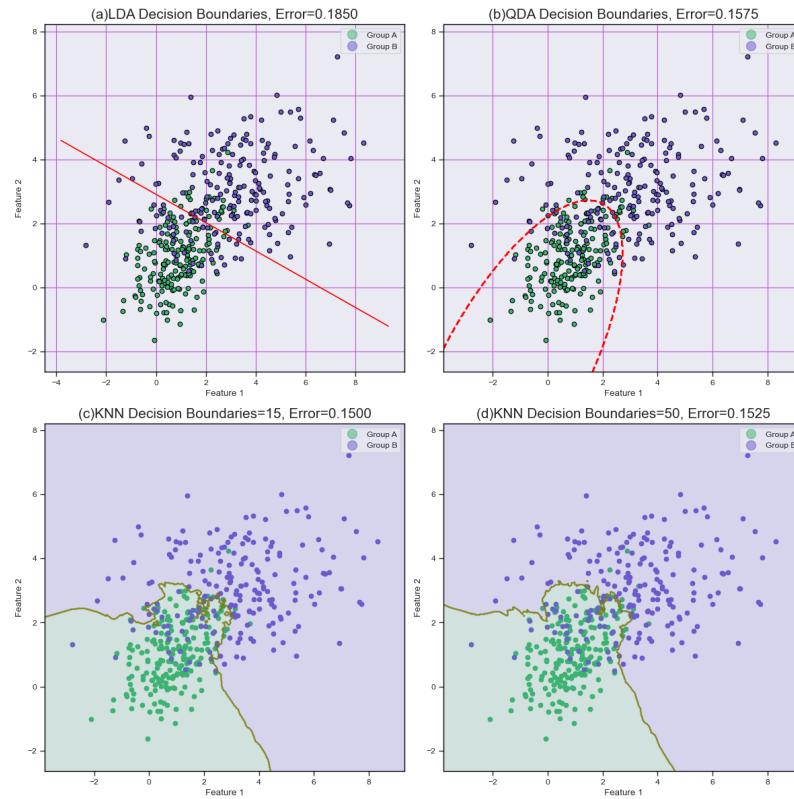


圖 5.5: 共變數矩陣不同正相關下 LDA,QDA,KNN=15 與 KNN=20 比較圖

各有千秋，但重點放到在意的預測錯誤率時，K=20 明顯優於 K=15，因此我們可以說這組資料中 K=20 是更加優秀的選擇，而總體來說 QDA 是最佳的學習器。我們從圖 5.6 可以看出，資料是向中央集中的，所以預測出的結果也沒有之前來的漂亮。

表 5.3: 不同共變數矩陣相關性一正一負下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=15)	KNN(K=20)
原資料錯誤率	0.2025	0.2025	0.2050	0.1925
訓練錯誤率	0.2093	0.2062	0.1937	0.1985
預測錯誤率	0.1999	0.1657	0.2249	0.1976

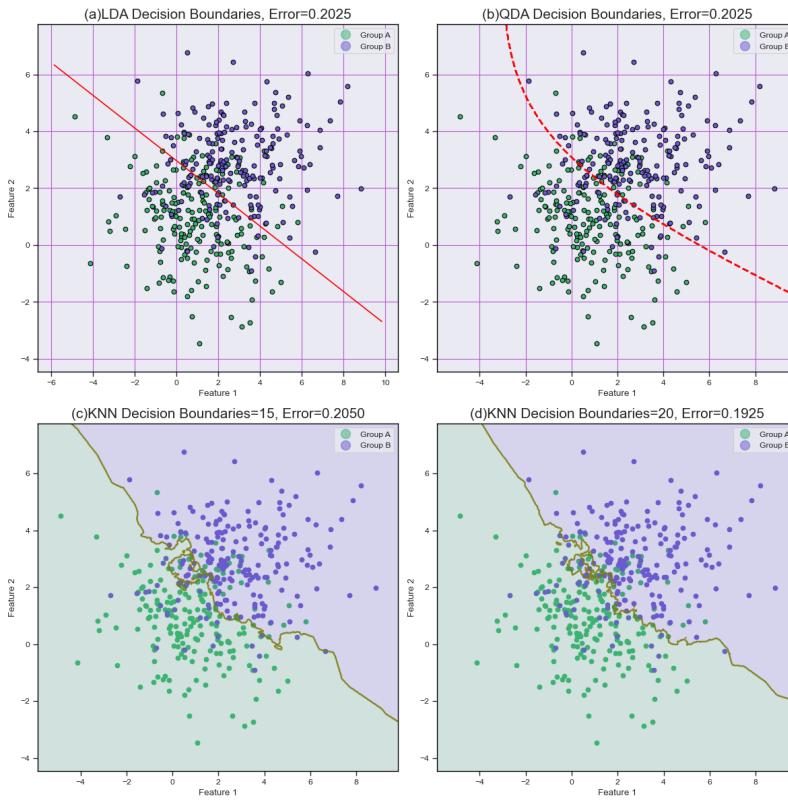


圖 5.6: 共變數矩陣不同且相關性一正一負下各學習器比較圖

5.4.4 兩群資料共變數不同且負相關

狀況四生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 3 \\ 0.5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 3 & -1 \\ -1 & 3 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 2 \\ 5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 5 & -0.2 \\ -0.2 & 4 \end{pmatrix}$$

由圖 5.7 與表 5.4，可以看出在不同負相關的共變數矩陣下，QDA 依然表現良好，且預測錯誤率上也是最優秀的選擇。而 $K=15$ 也是優於 $K=20$ 的結果，可以知道這組資料，在 $K=15$ 時模型配飾良好，且預測能力也比較高。

表 5.4: 不同共變數矩陣且負相關在不同學習器下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=15)	KNN(K=20)
原資料錯誤率	0.1225	0.0950	0.1075	0.1125
訓練錯誤率	0.1218	0.1031	0.1187	0.1156
預測錯誤率	0.1125	0.0625	0.0999	0.075

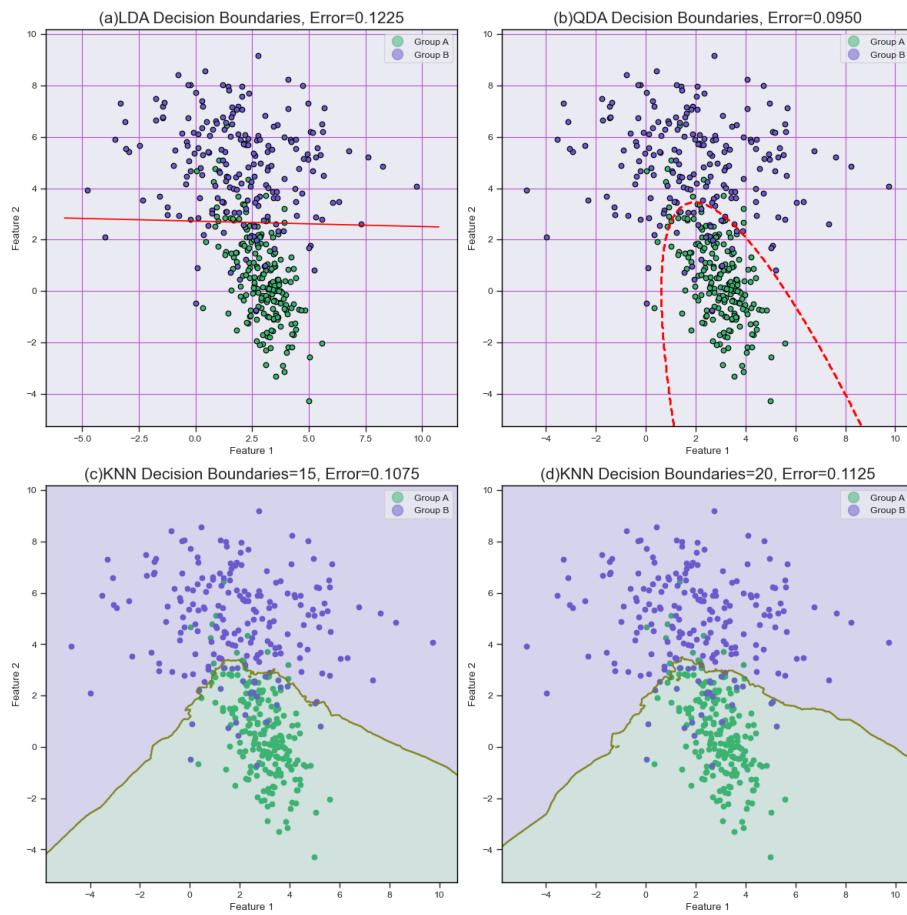


圖 5.7: 共變數矩陣不同且負相關下各學習器比較圖

5.4.5 兩群平均數接近且變異數不同

狀況五生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 0.5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 3 & -1 \\ -1 & 3 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 2 \\ 1.5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 5 & -0.2 \\ -0.2 & 4 \end{pmatrix}$$

由圖 5.8 與表 5.5，明顯看出因為平均值十分接近，因此資料過於集中，使得學習器的表現錯誤率都不太理想，唯一比較可以接受的只有 K=50 這個學習器，但他的預測錯誤率也是不太理想的結果，因此可能要用其他方法分析該組資料較佳。

表 5.5: 不同共變數矩陣且平均數接近在不同學習器下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=15)	KNN(K=50)
原資料錯誤率	0.2800	0.2644	0.2533	0.2467
訓練錯誤率	0.2472	0.2472	0.2500	0.2163
預測錯誤率	0.4000	0.3888	0.3444	0.3263

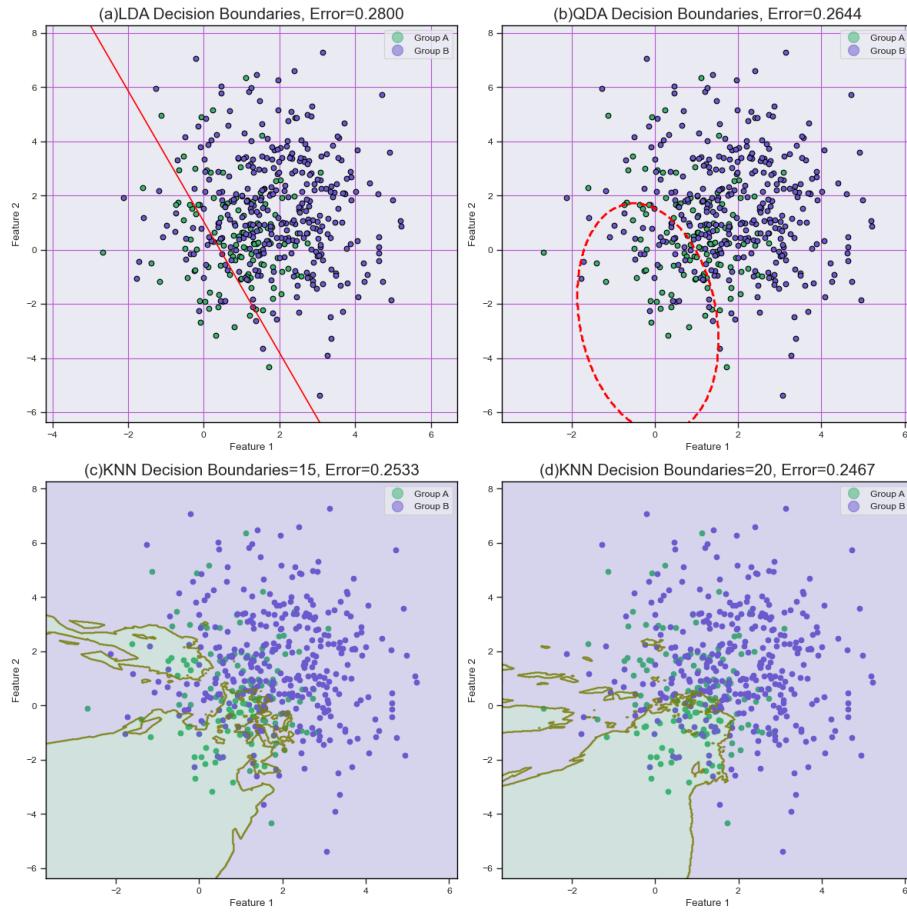


圖 5.8: 共變數矩陣不同且正相關下個學習器比較圖

5.4.6 兩群資料小結

根據上述整理我們可以得到表 5.6 與表 5.7，我們可以清楚地看出不同情況下哪些學習器是可用的哪些是不佳的。

- 狀況一：兩共變數矩陣相同
- 狀況二：兩共變數矩陣不同正相關
- 狀況三：兩共變數矩陣不同一正一負
- 狀況四：兩共變數矩陣不同負相關

- 狀況五：平均數接近

表 5.6: 兩群預測資料誤判率比較

誤判率	分析模型			
	LDA	QDA	KNN(K=15)	KNN(K=50)
狀況一	✓		✗	✗
狀況二	✗	✓		
狀況三		✓	✗	
狀況四	✗	✓		
狀況五	✗			✓

表 5.7: 兩群訓練資料誤判率比較

誤判率	分析模型			
	LDA	QDA	KNN(K=15)	KNN(K=50)
狀況一	✓			✗
狀況二	✗		✓	
狀況三	✗		✓	
狀況四	✗	✓		✓
狀況五			✗	✓

由上述結果我們可以得到幾個心得：

1. 只要符合 LDA 對共變數矩陣的假設如狀況一，該學習器都可以表現良好。
2. 共變數矩陣的正相關或負相關如狀況二與三，對於學習器的錯誤率判斷影響是不顯著的。
3. 如果讓平均值太過靠近容易讓資料混合再一起，導致錯誤率升高，就不適合使用這三種學習器了。

4. KNN 中的 K 值沒有絕對的好壞如狀況五，需要在不同資料中多加嘗試找出最適合的 K 值。

5.5 三群資料學習器評比

三群的生成資料一樣來自多變量常態母體，本章節會著重在共變數矩陣，對學習器有何影響進行不同的假設來進行評比。並將訓練資料與測試資料為 8:2 且進行 Bootstrapping 100 次計算平均誤判率。

5.5.1 三群共變數矩陣相同

狀況一生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

$$n_3 = 200, \quad \boldsymbol{\mu}_3 = \begin{pmatrix} 7 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

由圖 5.9 與表 5.8，可以看出因為資料分成三群所以原資料錯誤率都相同，但如果是看預測與訓練錯誤率的話，還是可看出在相同共變數矩陣下 LDA 優於 QDA。而在 KNN 下 K=5 也比 K=50 還來得好。

表 5.8: 三群：相同共變數矩陣在不同學習器下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=5)	KNN(K=50)
原資料 Error	0.0300	0.0300	0.0300	0.0333
訓練 Error	0.0306	0.0322	0.0280	0.0337
預測 Error	0.0296	0.0336	0.0303	0.0346

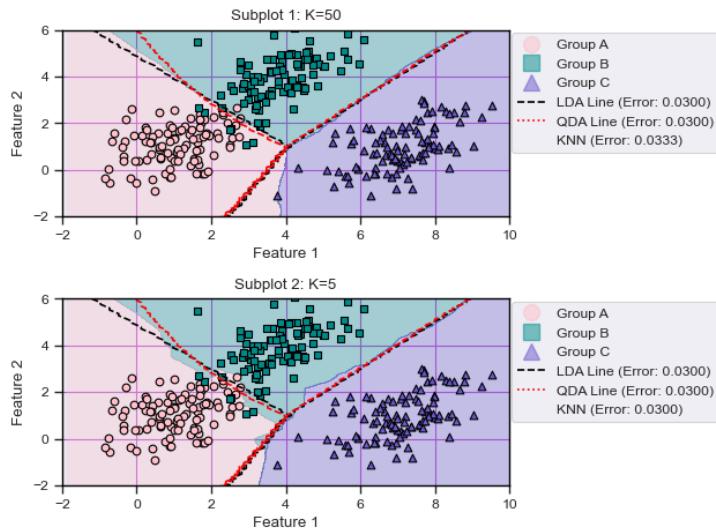


圖 5.9: 不同種學習器：三群共變數矩陣相同

5.5.2 三群兩組共變數相同且正相關

狀況二生成樣本數據：

$$\begin{aligned}
 n_1 &= 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} \\
 n_2 &= 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} \\
 n_3 &= 200, \quad \boldsymbol{\mu}_3 = \begin{pmatrix} 7 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 3 & 1 \\ 1 & 4 \end{pmatrix}
 \end{aligned}$$

由圖 5.10 與表 5.9，可以看出因為 LDA 是以直線呈現，所以其實跟圖 5.9 有些許的雷同，而也明顯看出如果失去假設 QDA 會優於 LDA。而當 K=5 時明顯會優於 K=50，是更加優秀的選擇。

表 5.9: 三群：兩組相同共變數矩陣且正相關在不同學習器下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=5)	KNN(K=50)
原資料 Error	0.0567	0.0500	0.0467	0.0567
訓練 Error	0.0555	0.0500	0.0468	0.0573
預測 Error	0.0570	0.0531	0.0515	0.0610

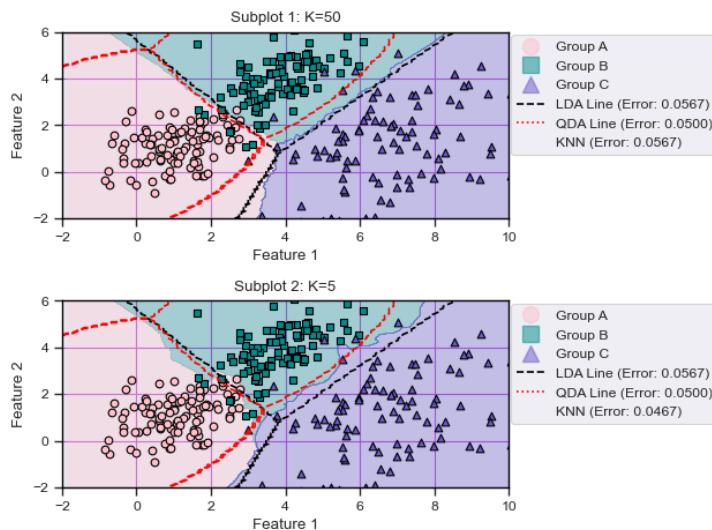


圖 5.10: 不同種學習器：三群共變數矩陣兩個相同且正相關

5.5.3 三群兩組共變數相同且負相關

狀況三生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$n_3 = 200, \quad \boldsymbol{\mu}_3 = \begin{pmatrix} 7 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 4 & -1 \\ -1 & 2 \end{pmatrix}$$

由圖 5.11 與表 5.10，可以看出不同的 K 值在訓練與預測上各有特色，但 K=50 在預測錯誤率方面是更加優秀的，因此選擇會偏向 K=50 的狀況。

表 5.10: 三群：兩組相同共變數矩陣且負相關在不同學習器下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=5)	KNN(K=50)
原資料 Error	0.0767	0.0567	0.0567	0.0767
訓練 Error	0.0797	0.0610	0.0534	0.0763
預測 Error	0.0811	0.0643	0.0785	0.0639

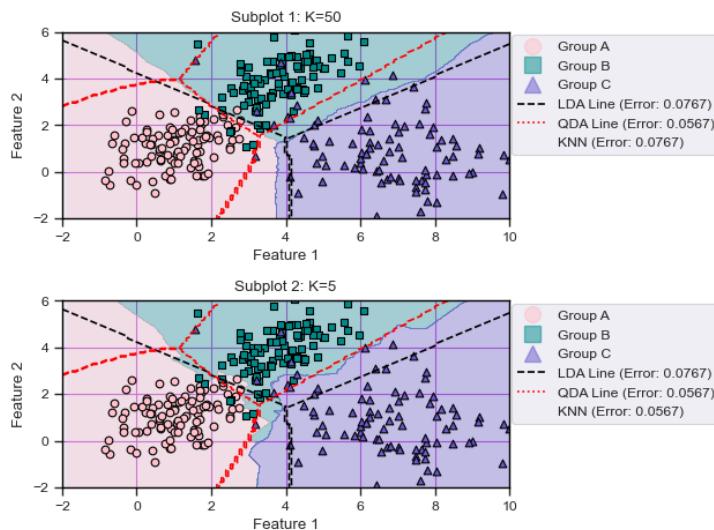


圖 5.11: 不同種學習器：三群共變數矩陣兩個相同且負相關

5.5.4 三群共變數皆不同

狀況四生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 5 & 2 \\ 2 & 6 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 0.6 & 0.5 \\ 0.5 & 0.9 \end{pmatrix}$$

$$n_3 = 200, \quad \boldsymbol{\mu}_3 = \begin{pmatrix} 7 \\ 2 \end{pmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 4 & -1 \\ -1 & 2 \end{pmatrix}$$

由圖 5.12 與表 5.11，在三種共變數矩陣不一樣的條件下，QDA 是表現最好的，也間接驗證假設是可行的。而在 KNN 的比較中 K=5 實在預測或訓練上都是不錯的選擇。

表 5.11: 三群：共變數矩陣皆不同在不同學習器下的誤判率

誤判率	分析模型			
	LDA	QDA	KNN(K=5)	KNN(K=50)
原資料 Error	0.1600	0.1067	0.1033	0.1400
訓練 Error	0.1585	0.1078	0.1109	0.1468
預測 Error	0.1598	0.1111	0.1378	0.1498

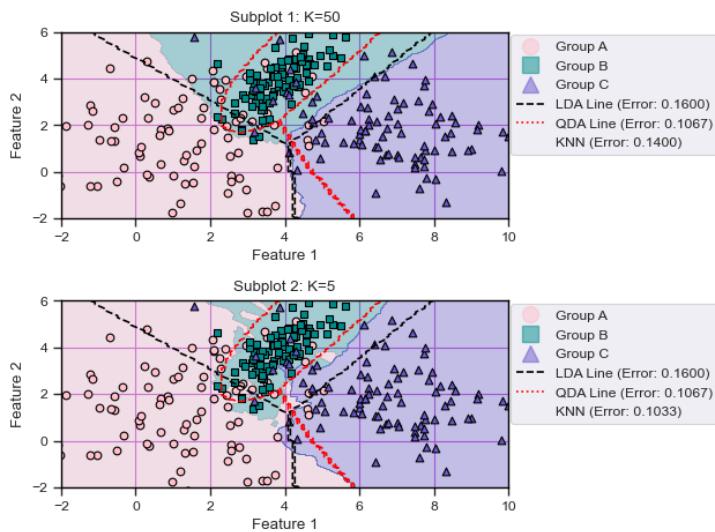


圖 5.12: 不同種學習器：三群共變數矩陣兩個相同且負相關

5.5.5 三群資料小結

根據上述整理我們可以得到表 5.12 與表 5.13，我們可以清楚地看出不同情況下哪些學習器是可用的哪些是不佳的。

- 狀況一：三共變數矩陣相同
- 狀況二：三共變數矩陣兩組相同且正相關
- 狀況三：三共變數矩陣兩組相同且負相關
- 狀況四：三共變數矩陣皆不同不同

由上述結果我們可以得到幾個心得：

1. 只要符合 LDA 對共變數矩陣的假設如狀況一，該學習器都可以表現良好。
2. 考慮到只有一組共變數矩陣不同的情況如狀況二與三，這時 QDA 不像兩群時來的優秀，反而是 KNN 更加適合處理該資料。

表 5.12: 三群預測資料誤判率比較

誤判率	分析模型			
	LDA	QDA	KNN(K=5)	KNN(K=50)
狀況一	✓			✗
狀況二		✓		✗
狀況三	✗			✓
狀況四	✗	✓		

表 5.13: 三群訓練資料誤判率比較

誤判率	分析模型			
	LDA	QDA	KNN(K=5)	KNN(K=50)
狀況一		✓		✗
狀況二		✓		✗
狀況三	✗		✓	
狀況四	✗	✓		

3. 四種狀況中訓練錯誤率 K=5 有三個都是最佳的，但預測錯誤率卻只有一個，說明預測能力不佳的問題。
4. KNN 中的 K 值沒有絕對的好壞，需要在不同資料中多加嘗試找出最適合的 K 值。

5.6 結論

經過這次的學習，了解到監督式學習可以以機率的角度去探討，並認識三種不同的學習器 LDA、QDA 與 KNN。熟悉他們運作的原理，並以實際資料做實驗，驗證他們的假設與背後的理論。在實際應用中，了解各學習器的優勢及適用情境，有助於選擇最適合解決特定問題的模型，對未來的事物上可以更容易加以運用。

第 6 章

監督式類神經網路學習器

本章討論監督式學習中類神經網路（Artificial Neural Network, ANN），解決複雜問題和圖像辨識方面有優越性能。本章會以前饋式（Feed-forward）類神經網路製作機械手臂以及手寫資料進行圖形辨識，來觀察當給定不同參數下，神經網路的呈現是如何與各自的準確率做比較。

6.1 類神經網路介紹

6.1.1 類神經網路基本概念

類神經網路是以電腦來模擬類似模仿神經結構的人工 AI，透過多層次的神經元網絡學習複雜的圖像特徵，可應用於影像分類、辨識等領域。其中主要組成分別為輸入層（Input Layer）、隱藏層（Hidden Layer）、輸出層（Output Layer）與權重（Weights）如圖 6.1¹。

1. 輸入層：接收外部輸入信息的一層，每個神經元對應著輸入的一個特徵。
2. 隱藏層：位於輸入層和輸出層之間，通常包含多個神經元。這些層在不同的模型中可以有不同數量和結構。且當層數達到

¹參考網站：<https://aws.amazon.com/tw/what-is/neural-network/>

兩層以上的神經網路，可以稱此為深度神經網路（Deep Neural Network,DNN）。

3. 輸出層：給出人工神經網路所有資料處理的最終結果，可以有一個或多個節點。
4. 權重：每條連接線上都有一個權重，決定了信號在神經網路中傳遞的強度。

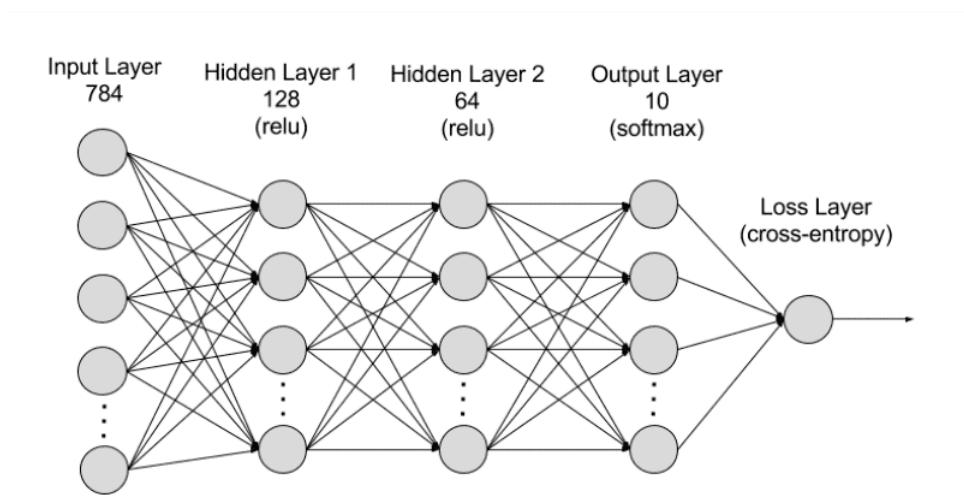


圖 6.1: 神經網路架構示意圖

6.1.2 前饋式類神經網路

前饋神經網路（Feedforward Neural Network）為人工智慧領域中，最早發明的簡單人工神經網路類型。其神經網路的特點在於，參數從輸入層向輸出層單向傳播，且所有節點都完全連結但流向不循環，且架構上比起 DNN 隱藏層少且簡單，如圖 6.2²。最左邊為輸入層，假設觀察到的解釋變數有三筆，中間的為隱藏層標示為 J,K 與 L, 此為具有三個隱藏層的前饋式類神經網路。其中隱藏層與節點個數會與不同模型訓練而有所改變，隱藏層的數目代表該模型預測的複雜程度，過少可能

²參考網站：<https://nordvpn.com/zh-tw/blog/shenjing-wang-lu/>

會導致預測能力較差。最右邊為輸出層輸出一個變數。接下來解釋各

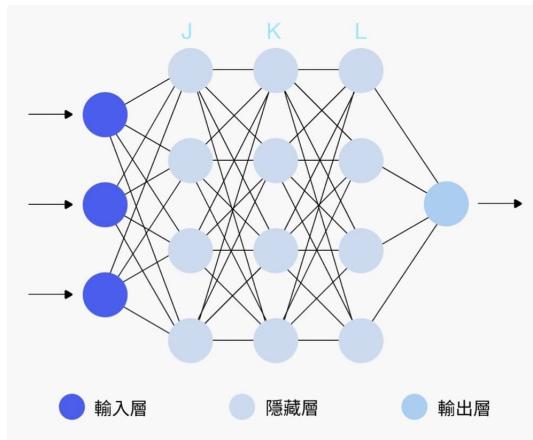


圖 6.2: 前饋式類神經網路架構圖

步驟關聯³，假設輸入層有 p 個變數 x_1, x_2, \dots, x_p ，輸出層有 r 個變數 $\hat{y}_1, \hat{y}_2, \dots, \hat{y}_r$ ，隱藏層的結構與所含的神經元數量 q ，則前饋式類神經網路的輸出與輸入間的數學關係寫成

$$\hat{y}_k = \sum_{i=1}^q \omega_{ki}^2 g\left(\sum_{j=1}^p \omega_{ij}^1 x_j + b_i^1\right) + b_k^2, \quad 1 \leq k \leq r \quad (6.1)$$

其中函數 $g(\cdot)$ 可以選擇為 $(-1 \leq g(z) \leq 1)$

$$g(z) = c_1 \frac{1 - e^{-c_2 z}}{1 + e^{-c_2 z}} \quad (6.2)$$

式 (6.1) 中的 ω_{ij}^1 與 b_i^1 代表第一個隱藏層的第 i 個神經元與 j 個輸入的權重係數 (Weightings) 與位階係數 (Biases)。同樣地 ω_{ki}^2 與 b_k^2 則是第二層的第 k 個神經元與前一隱藏層的第 i 個輸出的權重係數與位階係數。類神經網路根據其輸入資料 $x_j(n)$ 與 $y_k(n)$ 去調整 $\omega_{ij}^1, b_i^1, \omega_{ki}^2, b_k^2$ 使 $y_k(n)$ 可以與真實值 $\hat{y}_k(n)$ 的誤差為最小。假設共有 N 筆真實資料，則所謂類神經網路的訓練階段，寫成多變量函數的最小值問題，即式

³參考網站：<https://ntpuccw.blog/python-in-learning/sml-lesson-4-artificial-neural-network-ann-deterministic-approach/>

(6.3)

$$\begin{aligned}
 \min_{\Omega} e(\Omega) &= \min_{\Omega} \sum_{n=1}^N \sum_{k=1}^r (y_k(n) - \hat{y}_k(n))^2 \\
 &= \min_{\Omega} \sum_{n=1}^N \sum_{k=1}^r (y_k(n) - \sum_{i=1}^q \omega_{ki}^2 g(\sum_{j=1}^p \omega_{ij}^1 x_j + b_i^1) + b_k^2)
 \end{aligned} \tag{6.3}$$

其中參數集 $\Omega = \omega_{ij}^1, \omega_{ki}^2, b_i^1, b_k^2, i = 1, \dots, q, j = 1, \dots, p, k = 1, \dots, r$, 參數總共有 $pq + qr + q + r$ 個參數。因此類神經網路可視為一個非常複雜、非線性程度很高到函數，能將輸入 X 與輸出 Y 的關係配適的很完美。

6.2 機械手臂簡介與實測

本節利用類神經網路來測試機械手臂模型與其準確率。

6.2.1 逆運動方程式

簡單來說是預設所要達到的位置，所要設置關節參數的過程。以圖 6.3 為例，一般來說給出 θ_1 與 θ_2 而推出相對應位置 (X, Y) 座標，而逆運動方程式則是先給指定的位置 (X, Y) ，逆推求得 θ_1 與 θ_2 角度。

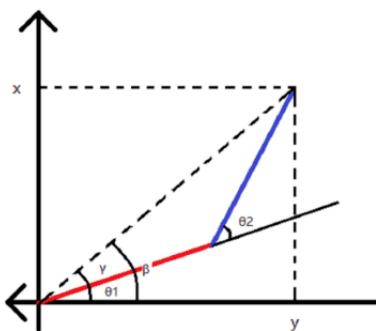


圖 6.3: 逆運動方程式座標平面圖

因此我們可以得出式 (6.5) 為逆運動方程式的解。

$$FAK : \begin{cases} x = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) \\ y = l_1 \sin(\theta_1) + l_2 \sin(\theta_2) \end{cases} \quad (6.4)$$

$$IKE : \begin{cases} \theta_1 = \arctan\left(\frac{y}{x}\right) - \arctan\left(\frac{l_2 \sin(\theta_2)}{l_1 + l_2 \cos(\theta_2)}\right) \\ \theta_2 = \arccos\left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2}\right) \end{cases} \quad (6.5)$$

但當維度升高後或角度變多時，會變得過於複雜，因此我們利用類神經網路的方式來解決。圖 6.4 為機械手臂範圍 (藍色) 與生成資料點 (黃色)。

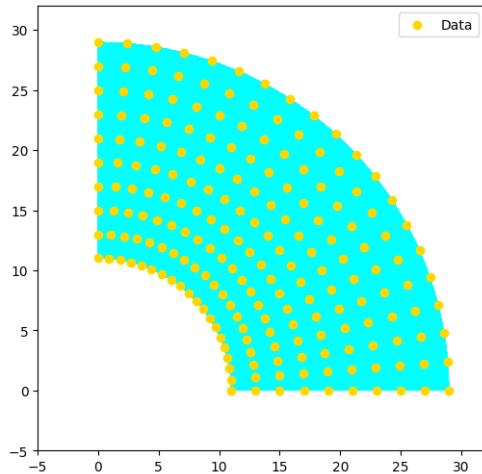


圖 6.4: 機械手臂位置圖範例

6.2.2 機械手臂實作

接著使用 sklearn 套件 MLPRegressor, 進行類神經網路的模型訓練，自訂神經元個數與運用遞迴的方式運作。本節將探討如果隱藏層中有不同個數神經元，而真實位置與預測位置會有怎麼差異，並算出均方誤差 (Rmse) 來數據化，觀察模型擬和值與真實值的差異。圖 6.5 我們分別設定 5,10,15 與 20 四組不同個數的神經元來做比較。其中均方根誤

差 (Rmse) 就是殘差的標準差。殘差是離迴歸線資料點有多遠的測量值 RMSE, 則是這些殘差值有多散開的測量值。

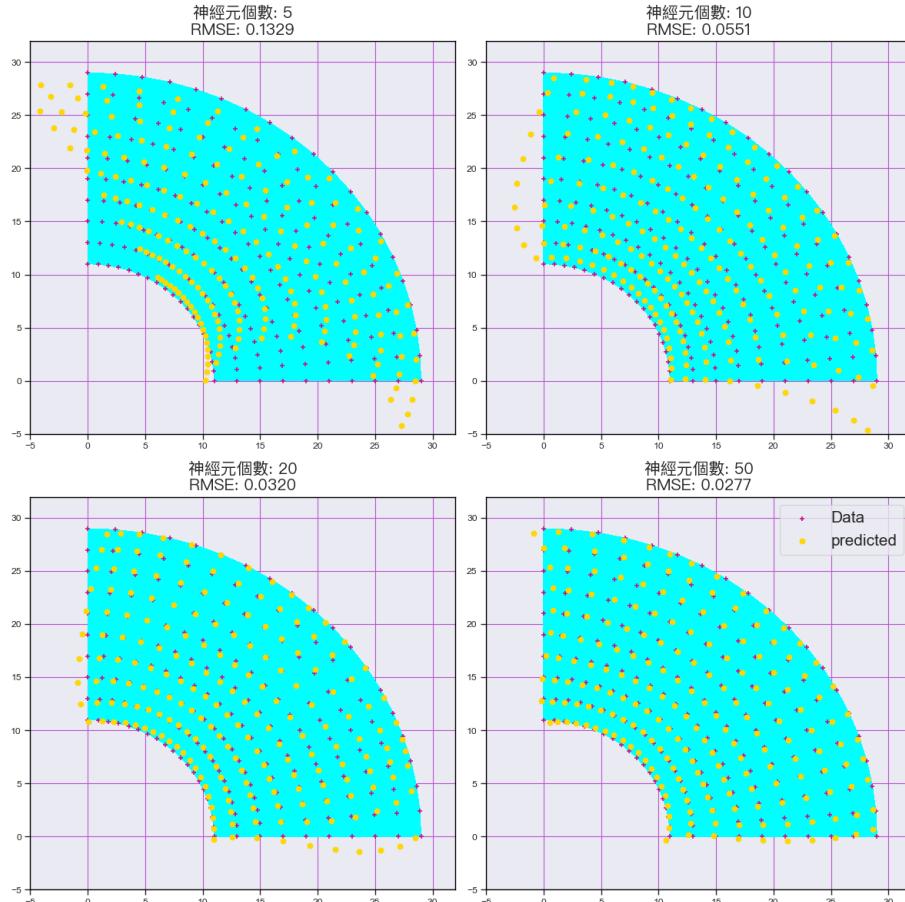


圖 6.5: 不同神經元數目下的 Rmse 比較圖

由圖 6.5 我們可以看出當神經元數目越多，相對 Rmse 值越小，這代表擬合結果越好。但真的隱藏層數目越多數值就會越好嗎？如圖 6.6 可以看出雖然 30 的數目比較小，但所生成的 Rmse 是比較好的，沒有神經元比較多就比較好，不同資料的情況下，有不同的最佳數目的值。接下來將運用不同演算法來製作，分別為 lbfgs, sgd 與 adam，來看在設定 10 個神經元的情況下，個別的 Rmse 值是如何。圖 6.7 可以發現在設定神經元數目為 10 得情況下，lbfgs 演算法下所跑出的 Rmse 值是最

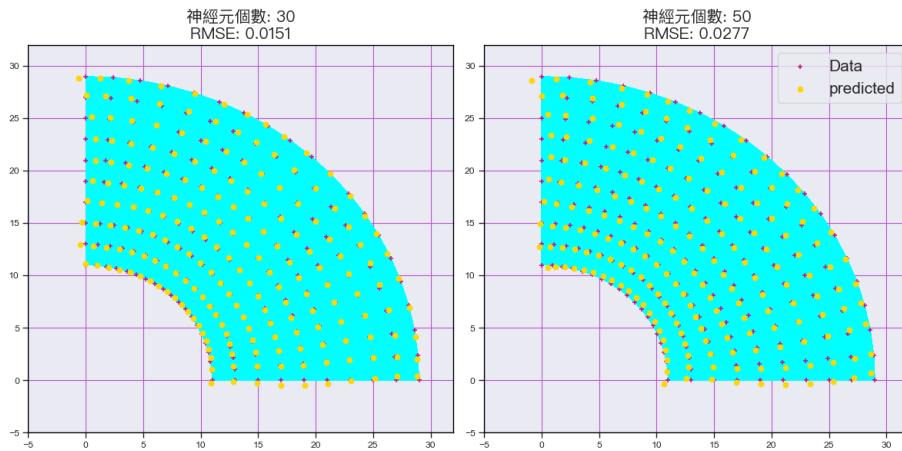


圖 6.6: 神經元數目 vs Rmse

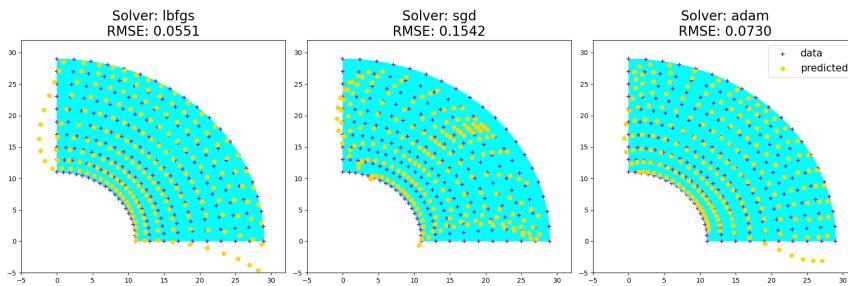


圖 6.7: 神經元 10 個下 lbfgs,sgd 與 adam 比較圖

佳的。

6.2.3 小節

根據上述圖 6.5 與圖 6.6 我們可以得到表 6.1。由表 6.1 可以看出當隱

表 6.1: 機械手臂隱藏層數目 vs 模型預測

	隱藏層數目				
	5	10	20	30	50
Rmse	0.1329	0.0551	0.0320	0.0151	0.0227

藏層神經元增加時，Rmse 確實會比較優秀，間接證明越複雜的模型需

要的隱藏層神經元越多。但不是越多一定是最好的，不同資料最佳的數量都不同。

6.3 資料生成與預測

6.3.1 訓練與測試資料生成

當生成資料太過均勻規律時，會導致預測能力不佳，我們應該要生成一個更加隨機與密集的資料，來增加預測能力。如圖 6.8 我們利用亂數生成，並將資料拆成訓練與測試資料。

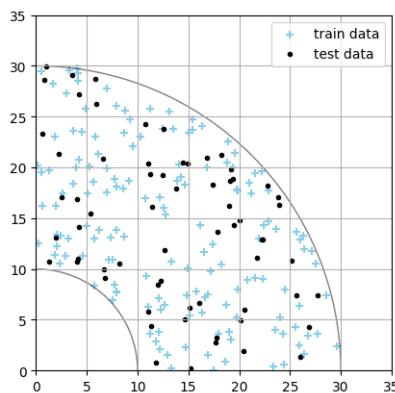


圖 6.8: 訓練與測試資料生成

6.3.2 神經元個數與樣本個數

前一節提到了樣本生成，接下來將套入類神經網路中，我們將比較不同神經元數目與樣本個數，並計算出其 SSE 值進行比較，其中 SSE 為殘差平方和，用於測量隨機誤差或未解釋的變異。由圖 6.9 可以看出在隱藏層增加樣本數不變的情況下，SSE 的值是變好的，跟前面機械手臂一樣增加隱藏層是可以將模型訓的變好的方法。而當隱藏層不變樣本數增加的情況下，SSE 的值也是有變好，但相較上一個條件卻是不那麼顯著的成長。而當兩個條件都增加時，我們也可以明顯看出 SSE

變更優秀。而圖 6.10 我們生成圖 6.9 的訓練誤差圖趨勢圖。

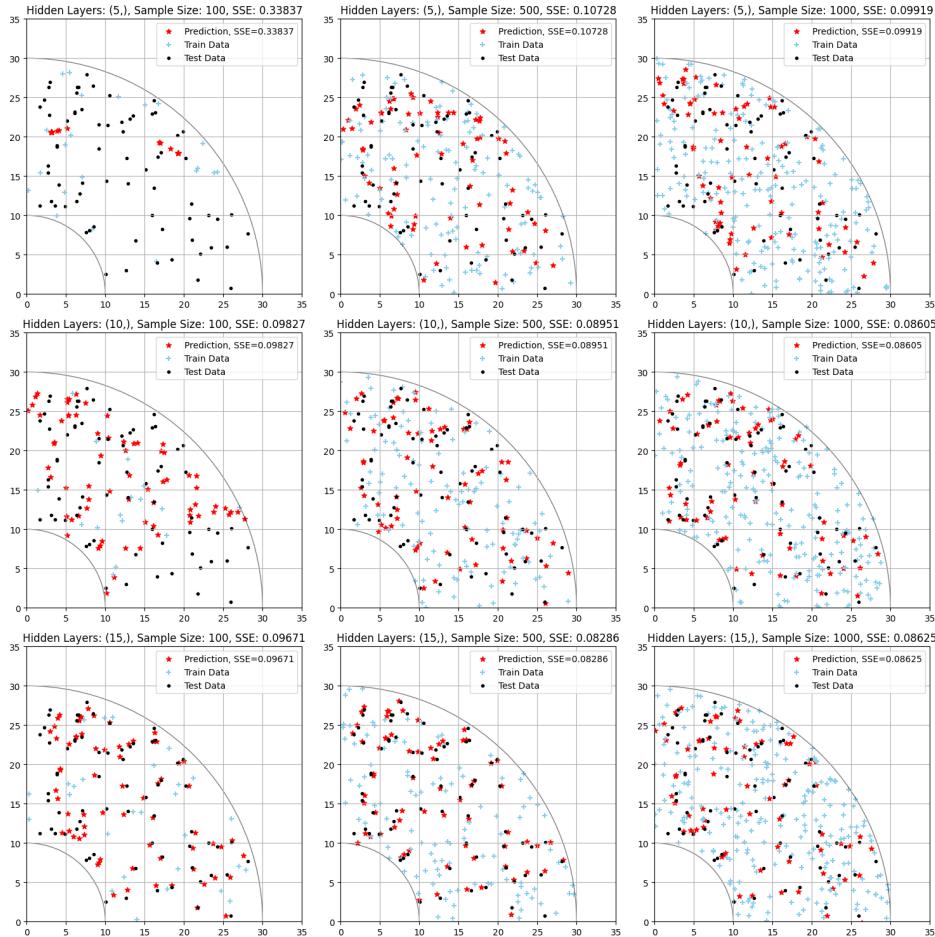


圖 6.9: 資料樣本數目 (100,500,1000) vs 神經元數目 (5,12,15)

由圖 6.9 與圖 6.10 可以看出當只改變樣本數目時每個訓練的成果差異都不會太明顯。但如果是改變隱藏層的數目時，可以明顯看出圖型的變化預測能力是增加的，尤其是多隱藏層與多樣本時圖型更加的平滑。

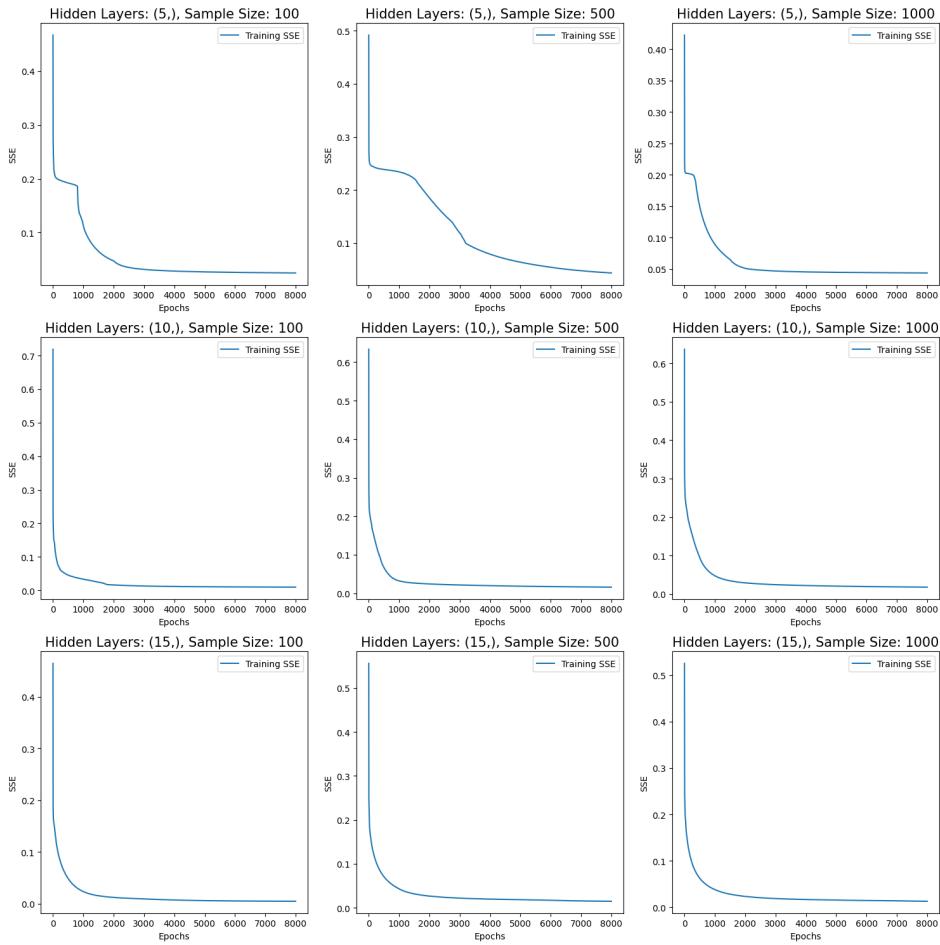


圖 6.10: 圖 6.9 對應訓練誤差趨勢圖

6.3.3 圖像辨識

類神經網路不僅僅只能用在機械手臂，也可以用在影像圖形辨識，本節會利用資料來做數字與英文的類神經網路判讀。因為每個人的字跡都不一樣，我們希望是可以透過神經網路的幫助來判讀，使其能對圖形的辨識能力做探討。圖 6.11 為一個 20×30 的蒙太奇圖陣，分別製作不同隱藏層下的混淆矩陣，測試誤差圖並計算出他預測的準確性。由圖 6.12 與 6.13 可以發現當你設定隱藏層只有 10 層時，他的準確率只有 0.8520，例如他將一些數字四判別成 5,6 和 9，對比於 30 層隱藏層的

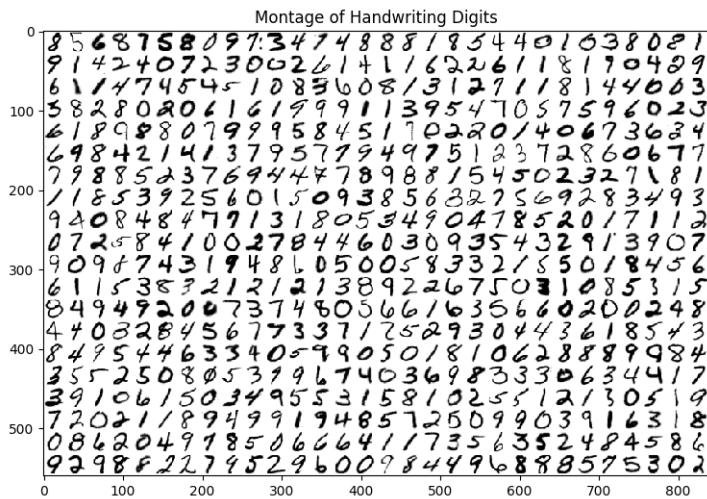


圖 6.11: 數字 20*30 蒙太奇圖陣

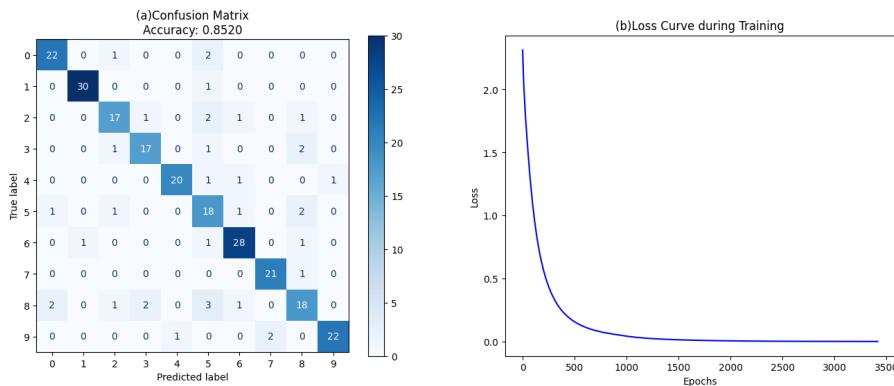


圖 6.12: 10 隱藏層混淆矩陣與測試誤差圖

圖，準確率是 0.9160 是明顯高於 10 層的。且兩邊的誤差圖可以看出，30 層的學習的速度是優於 10 層的。在從圖 6.12(a) 我們可以看出在數字 8 與 2 時特別容易出錯，錯誤率分別是 0.2272 與 0.333，而圖 6.13(a) 每個數字出錯都非常平均。由上述我們可以推測，在影像辨識上多一點隱藏層有助於影像判別。

接著我們將看英文字的圖刑判別，先看他的蒙太奇圖陣，如圖 6.14 為一個 20*30 英文字的蒙太奇陣圖，我也比照數字製作出他的混淆矩陣與測試誤差圖，並把隱藏層固定在 30。

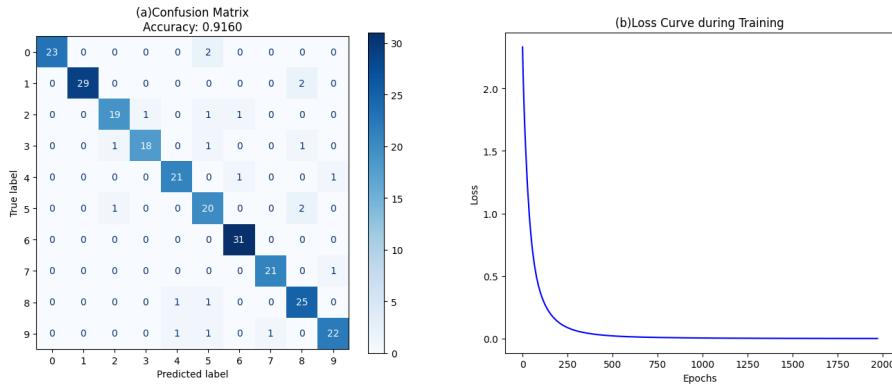


圖 6.13: 30 隱藏層混淆矩陣與測試誤差圖



圖 6.14: 英文字 20*30 蒙太奇圖陣

由圖 6.15 我們可以看出雖然都是 30 層的隱藏層，但相較於數字英文字的準確率只有 0.6416，是低非常多的，原因可能在於相較於數字只需判斷 10 個字，英文卻要判斷 26 個字母，所以對於判別來說較為複雜也較困難。因此我們應該再設多點的隱藏層可能就會好轉，如圖 6.16 將隱藏層設定為 50，準確率也上升到了 0.7080，但缺點就是執行時間就會比較久。

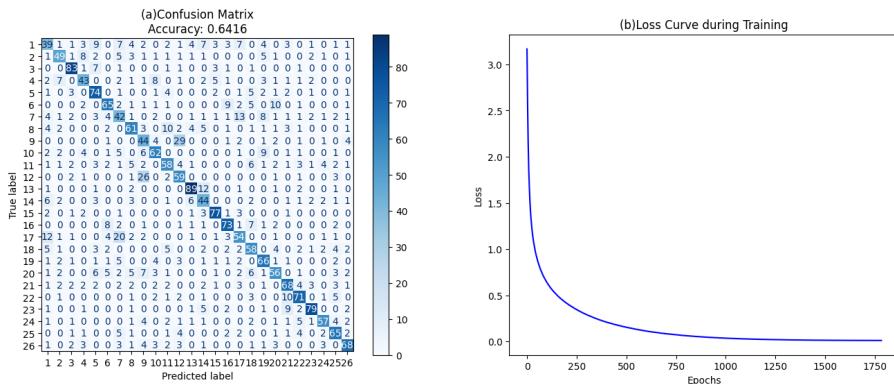


圖 6.15: 30 隱藏層英文混淆矩陣與測試誤差圖

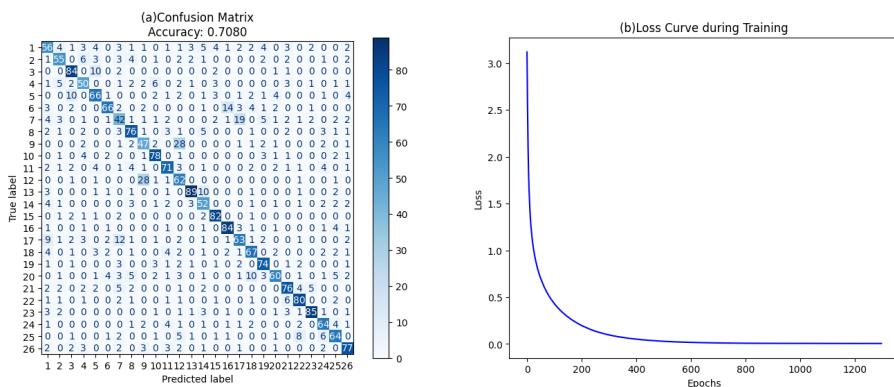


圖 6.16: 50 隱藏層英文混淆矩陣與測試誤差圖

在類神經網路的方法當中，以往都要操作很久的程式，而現在多了許多套件的幫忙省下不少時間，讓不管是機械手臂或影像辨識，都有所成長。類神經網路 AI 已經是未來的趨勢，需要更加了解如何運用並熟知其中的利弊，比起前章節類神經網路可以設定的參數更多，未來能發展的可能性也跟著增加。

在學習過程中，我們會遇到許多問題，例如：資料不足、過度訓練、欠訓練、過度複雜的模型等。這些問題可能會導致模型的效能不佳，甚至無法正確地進行預測。要解決這些問題，我們可以採取以下幾種方法：

- 資料不足：如果訓練資料量不足，模型可能無法學到足夠的特徵，導致泛化能力差。此時可以考慮擴充資料集、增加資料樣本數或是使用增強技術（如資料增廣）。
- 過度訓練：當模型在訓練資料上表現很好，但在驗證資料上表現卻變差時，表示模型可能已經過度訓練了。此時可以調整訓練步驟，減少訓練次數或增加正則化項。
- 欠訓練：如果訓練過程太短，模型可能無法學到足夠的特徵，導致泛化能力差。此時可以增加訓練步驟或資料樣本數。
- 過度複雜的模型：如果模型結構過於複雜，它可能會過度適應訓練資料，而在驗證資料上表現變差。此時可以考慮簡化模型結構、減少參數或是使用正則化技術。

總之，在學習過程中，我們需要不斷地調整和優化模型，以達到最佳的效能。

第 7 章

蒙地卡羅模擬實驗：學習器的評比

本章會評比前四、五與六章學習器的模型，有邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30，利用蒙地卡羅模擬實驗，檢測每個學習器在給定不同參數下，其訓練與預測的錯誤率會是如何，並進行各自的比較，選出不同種資料中最適合的學習器，更加瞭解每個學習器的特色。

7.1 兩群資料綜合學習器評比

本節將會用兩組多元常態分配生成資料，把資料分拆成 8:2 的訓練資料與預測資料，並運用不同的假設更改其中的參數，算出其錯誤率，進行學習器評比。

7.1.1 兩群資料共變數矩陣皆相同

狀況一生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 2 \\ 5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

表 7.1: 相同共變數矩陣在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.0779	0.0788	0.0791
預測錯誤率	0.0863	0.0869	0.0883

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.0771	0.0801	0.0799	0.0808
預測錯誤率	0.1066	0.1009	0.0895	0.0898

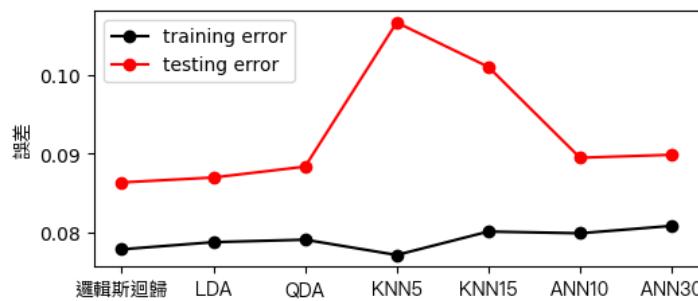
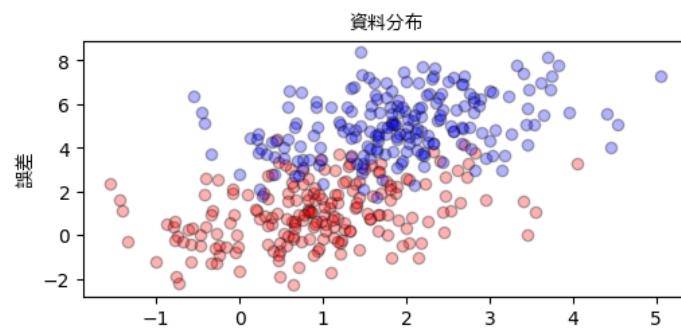


圖 7.1: 共變數矩陣相同下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

由圖 7.1 與表 7.6 可以看出當共變異數矩陣都相同時，訓練資料在 KNN(K=5) 是最好的而 ANN(A=30) 是最差的，但在預測資料時

KNN(K=5) 又是最差的反而是邏輯斯迴歸是最好的，可能的原因在於剛好資料配飾對 KNN(K=5) 有利所以訓練資料才會比較好。

7.1.2 兩群資料共變數矩陣不同且兩者正相關

狀況二生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 4 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

表 7.2: 不同共變數矩陣且正相關在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.2115	0.2125	0.1842
預測錯誤率	0.2151	0.2137	0.1958

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.1537	0.1773	0.1860	0.1831
預測錯誤率	0.2280	0.2112	0.2043	0.2016

由圖 7.2 與表 7.2 可以看出當共變異數矩陣都不同且兩變異數矩陣正相關時，訓練資料在 KNN(K=5) 是最好的而 LDA 是最差的，但在預測資料時 KNN(K=5) 又是最差的反而是 QDA 是最好的，可能的原因在於剛好資料配飾對 KNN(K=5) 有利所以訓練資料才會比較好。

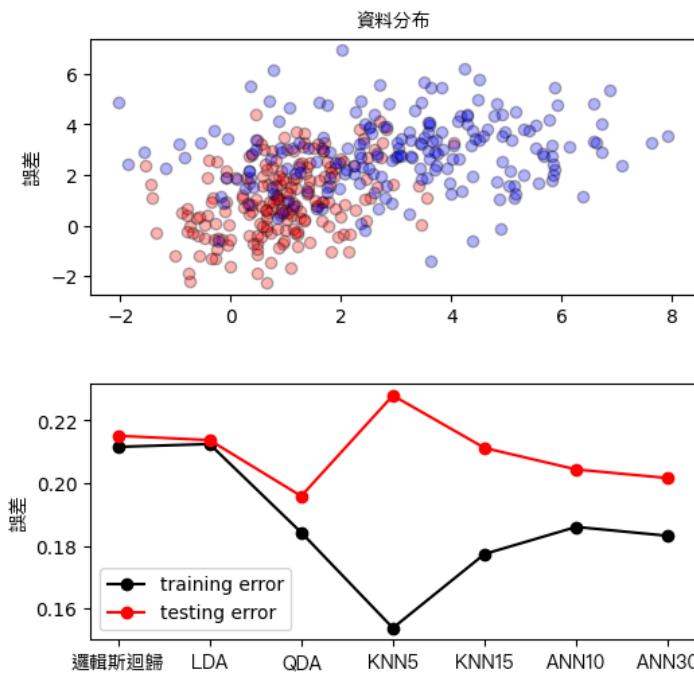


圖 7.2: 共變數矩陣不同且正相關下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

7.1.3 兩群資料共變數矩陣不同且矩陣關係一正一負

狀況三生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 3 & -0.5 \\ -0.5 & 2 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 4 & 0.2 \\ 0.2 & 2 \end{pmatrix}$$

表 7.3: 不同共變數矩陣且關係一正一負在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.1692	0.1684	0.1702
預測錯誤率	0.1848	0.1837	0.1867

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.1454	0.1643	0.1674	0.1699
預測錯誤率	0.2043	0.1934	0.1851	0.1874

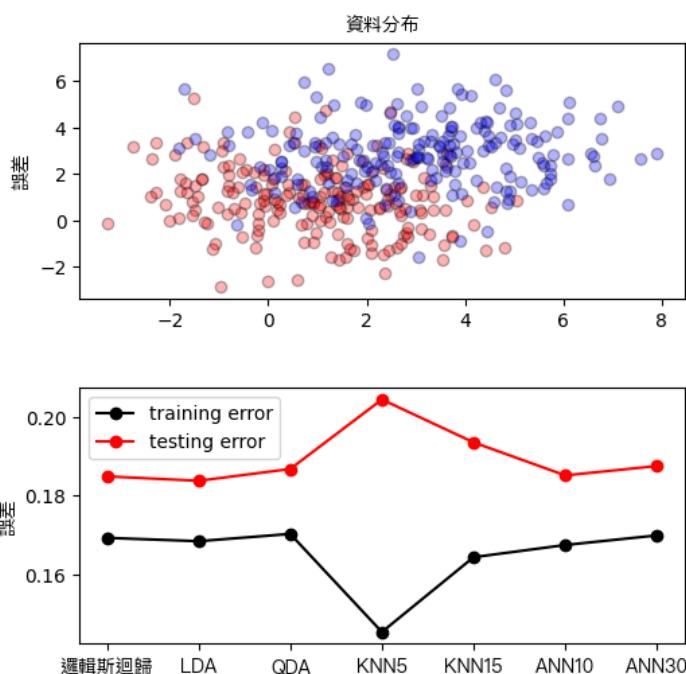


圖 7.3: 共變數矩陣不同且關係一正一負下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

由圖 7.3 與表 7.3 可以看出當共變異數矩陣都不同且兩變異數呈現一正一負關係時，訓練資料在 KNN(K=5) 是最好的而 QDA 是最差的，但在預測資料時 KNN(K=5) 又是最差的反而是 LDA 是最好的，可能的原因在於剛好資料配飾對 KNN(K=5) 有利所以訓練資料才會比較好。

7.1.4 兩群資料共變數矩陣不同且兩者負相關

狀況四生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 3 \\ 0.5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 3 & -1 \\ -1 & 3 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 2 \\ 5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 5 & -0.2 \\ -0.2 & 4 \end{pmatrix}$$

表 7.4: 不同共變數矩陣且負相關在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.1239	0.1245	0.0855
預測錯誤率	0.1271	0.1278	0.0882

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.0737	0.0899	0.1235	0.0852
預測錯誤率	0.1058	0.0991	0.1274	0.0920

由圖 7.4 與表 7.4 可以看出當共變異數矩陣都不同且兩變異數矩陣負相關時，訓練資料在 KNN(K=5) 是最好的而 LDA 是最差的，在預測資料時 QDA 是最好的而 LDA 還是最差的，可以推測這組資料不適合使用 LDA 分析。

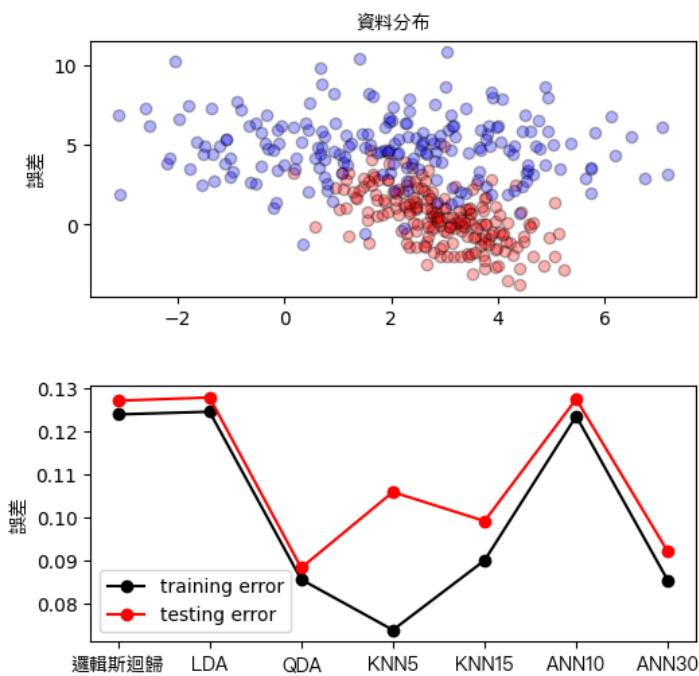


圖 7.4: 共變數矩陣不同且負相關下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

7.1.5 兩群資料平均數接近且共變數矩陣不同

狀況五生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 0.5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 3 & -1 \\ -1 & 3 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 2 \\ 1.5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 5 & -0.2 \\ -0.2 & 4 \end{pmatrix}$$

由圖 7.5 與表 7.5 可以看出當共變異數矩陣都不同且把平均設定相近時，訓練資料在 KNN(K=5) 是最好的而邏輯斯迴歸是最差的，在預測資料時 ANN(A=10) 是最好的而 KNN(K=15) 是最差的。

表 7.5: 不同共變數矩陣且平均數相近在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.2751	0.2733	0.2631
預測錯誤率	0.2836	0.2812	0.2741

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.1952	0.2458	0.2523	0.2550
預測錯誤率	0.2823	0.2851	0.2658	0.2744

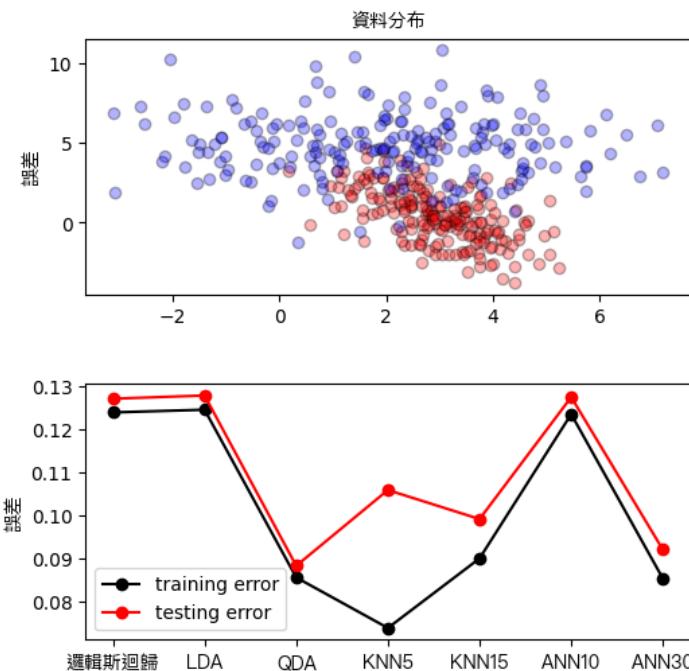


圖 7.5: 共變數矩陣不同且平均數相近下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

7.2 三群資料綜合學習器評比

本節將會用三組多元常態分配生成資料，把資料分拆成 8:2 的訓練資料與預測資料，並運用不同的假設更改其中的參數，算出其錯誤率，進行學習器評比。

7.2.1 三群資料共變數矩陣皆相同

狀況一生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

$$n_3 = 200, \quad \boldsymbol{\mu}_3 = \begin{pmatrix} 7 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

表 7.6: 相同共變數矩陣在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.0243	0.0260	0.0257
預測錯誤率	0.0293	0.0295	0.0295

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.0253	0.0259	0.0245	0.0239
預測錯誤率	0.0301	0.0303	0.0304	0.0290

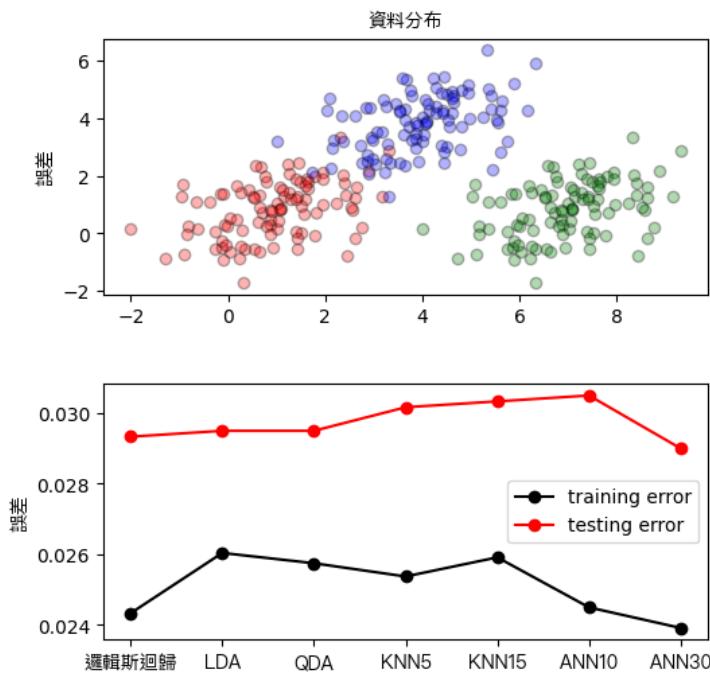


圖 7.6: 三群資料共變數矩陣相同下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

由圖 7.6 與表 7.6 可以看出當共變異數矩陣都相同時，訓練與預測資料在 ANN(A=30) 都是最好的，表示這組資料非常適合 ANN(A=30)，而 LDA 在訓練的時候是較差的，預測資料則是在 ANN(A=10) 是最差的。

7.2.2 三群資料共變數矩陣兩組相同且正相關

狀況二生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$n_3 = 200, \quad \boldsymbol{\mu}_3 = \begin{pmatrix} 7 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 3 & 1 \\ 1 & 4 \end{pmatrix}$$

表 7.7: 三群資料兩組共變數矩陣相同且正相關在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.0572	0.0739	0.0604
預測錯誤率	0.0551	0.0621	0.0581

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.0603	0.0674	0.0640	0.0619
預測錯誤率	0.0586	0.0586	0.0643	0.0614

由圖 7.7 與表 7.7 可以看出當共變異數矩陣都兩組相同且正相關時，訓練與預測資料在邏輯斯迴歸是最好的，表示這組資料非常適合邏輯斯迴歸，而 LDA 在訓練的時候是較差的，預測資料則是在 ANN(A=10) 是最差的。

7.2.3 三群資料共變數矩陣兩組相同且負相關

狀況三生成樣本數據：

$$n_1 = 200, \quad \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

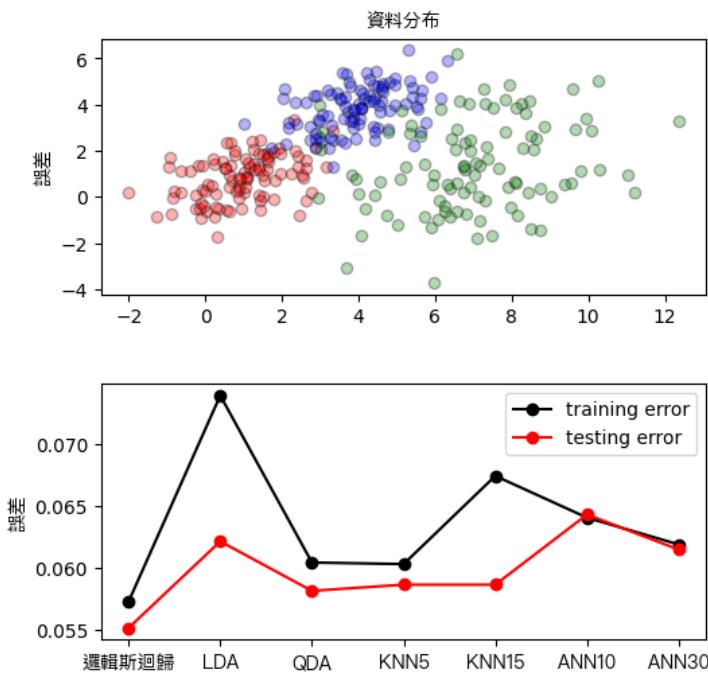


圖 7.7: 三群資料共變數矩陣兩組相同且正相關下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

$$n_3 = 200, \quad \boldsymbol{\mu}_3 = \begin{pmatrix} 7 \\ 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 4 & -1 \\ -1 & 2 \end{pmatrix}$$

表 7.8: 三群資料兩組共變數矩陣相同且負相關在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.0698	0.0907	0.0711
預測錯誤率	0.0726	0.0865	0.0733

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.0602	0.0765	0.0721	0.0711
預測錯誤率	0.0736	0.0808	0.0704	0.0711

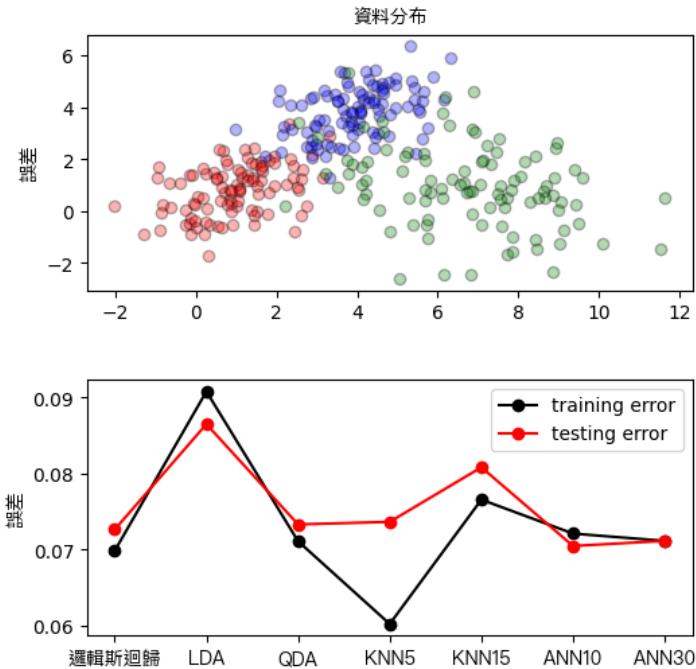


圖 7.8: 三群資料共變數矩陣兩組相同且負相關下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

由圖 7.7 與表 7.7 可以看出當共變異數矩陣兩組相同且負相關時，訓練與預測資料在 LDA 都是最差的，而 KNN(K=5) 在訓練的時候是較佳的，預測資料則是在 ANN(A=10) 是最好的。

7.2.4 三群資料共變數矩陣都不同

狀況四生成樣本數據：

$$n_1 = 200, \quad \mu_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \Sigma_1 = \begin{pmatrix} 5 & 2 \\ 2 & 6 \end{pmatrix}$$

$$n_2 = 200, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 0.6 & 0.5 \\ 0.5 & 0.9 \end{pmatrix}$$

$$n_3 = 200, \quad \boldsymbol{\mu}_3 = \begin{pmatrix} 7 \\ 2 \end{pmatrix}, \quad \boldsymbol{\Sigma}_3 = \begin{pmatrix} 4 & -1 \\ -1 & 2 \end{pmatrix}$$

表 7.9: 三群資料共變數矩陣皆不同在多個學習器下的誤判率

誤判率	分析模型		
	邏輯斯迴歸	LDA	QDA
訓練錯誤率	0.16325	0.1789	0.1350
預測錯誤率	0.1663	0.1761	0.1364

誤判率	分析模型			
	KNN(K=5)	KNN(K=15)	ANN=10	ANN=30
訓練錯誤率	0.1251	0.1479	0.1469	0.1462
預測錯誤率	0.1549	0.1538	0.1518	0.1504

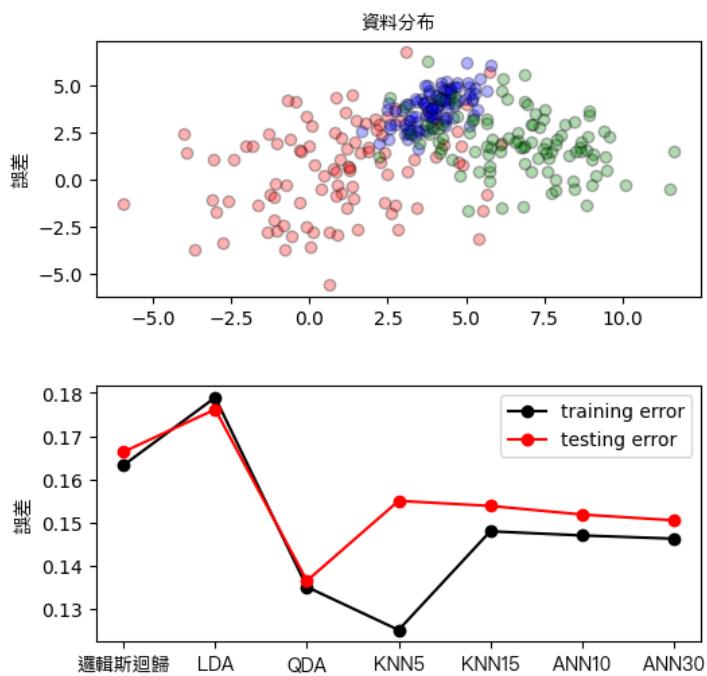


圖 7.9: 三群資料共變數矩陣皆不同下邏輯斯回歸模型、LDA、QDA、KNN=5、KNN=15、ANN=10、ANN=30 比較圖

由圖 7.9 與表 7.9 可以看出當共變異數矩陣兩組相同且負相關時，訓練與預測資料在 LDA 都是最差的，而 KNN($K=5$) 在訓練的時候是較佳的，預測資料則是在 QDA 是最好的。

7.3 結論

每一種學習器都有擅長處理的資料類型，例如:ANN 對於複雜的資料就處理得比其它的還好，所以要了解每個學習器的特性在適合得資料上，放入最合適的學習器，能夠增加可以精準的處理想要的資料問題的機率。希望在往後碩班研究上處理類似資料時，可以靈活的運用所學，不僅要快速解決問題，也可以精準的得到想要的數據。