國立清華大學

智慧化企業整合

期末報告

基於深度學習神經網路於心臟病確診分類之應用

指導教授：邱銘傳博士

組別：8

學生：陳譽升 109034538

一、 背景介紹

1. 情境描述

現今醫療體系面臨的挑戰，是如何在成本控制的考量下提供有品質的醫療服務，而有品質的醫療服務意味者病患能正確地被確診，以及接受最洽當的治療方案。而隨著近年精準醫學（Precision Medicine）興起，許多醫療機構及研究都投入於如何將個人的資料庫透過預測來輔佐決策，以提升臨床醫療的正確性。而本研究將從心臟病的預測模型出發，結合課程中學習的分類統計與深度學習神經網路方法進行模型的比較，衡量其成效。

2. 問題定義

上網搜尋心臟病確診預測資料集進行探討，發現 Kaggle 有關心臟病預測資料庫，決定以此作為本次報告之主題。而此資料及提供 14 項特徵來描述心臟病相關變數，從中挑出重要的特徵訓練模型並預測是否確診心臟病，以下將透過 5W1H 手法進行問題定義分析：

What?	心臟病確診預測
When?	欲了解自身是否罹患心臟病時
Who?	心臟不適之民眾與心臟病科醫生
Where?	醫院及診所
Why?	在成本控制下提供快速及精準之確診判斷，讓病患得以安心接受相關治療
How?	資料視覺化、機器學習、深度學習

表 1、臟病確診預測 5W1H 問題定義

3. 預測模型研究流程

下圖 1 為此預測模型之研究架構圖，首先將資料進行預處理，包含將類別型轉換為 Dummy Variable 與標準化，並依照資料相關性分析與視覺化之結果進行特徵篩選，以得到各特徵變數和目標變數，接著進行資料分割等處理作業。透過深度學習神經網路 NN 模型框架完成心臟病預測模型，然後為了使用模型有更好的成效，進行神經網路相關超參數調整，最後建立其他機器學習模型(羅吉斯回歸與隨機森林分類器)以驗證神網路模型之效度。

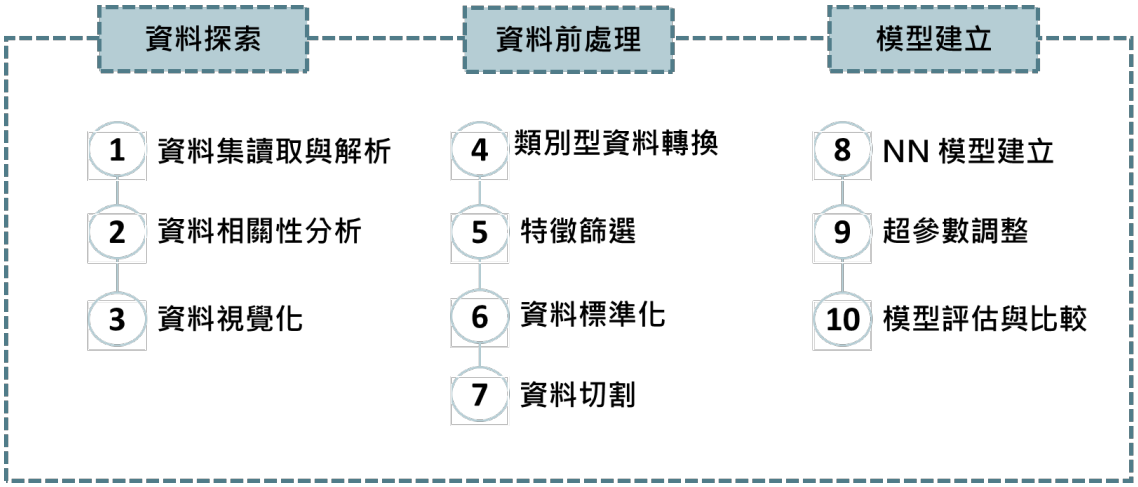

圖 1、心臟病預測模型研究架構

二、 資料探索

1. 資料集說明

此研究將使用 UCI Machine Learning Repository 所供心臟病預測資料庫，其數據共有 303 筆，14 項變數，其中目標應變數為：Target（1 為有心臟病、0 為無心臟病），其中 age、sex 為個人資料、cp（胸痛程度）為受測者自填項目，其餘皆為受測者透過檢查後而得的醫學相關數據。下表為其資料表類別與屬性表。其中經過 Python 讀取資料後，發現並無任何遺失值，透過 Summary( ) 函式以及檢視各變數分佈圖後，並無特定變數具有離群值（Outliers），故僅調整資料型態至其正確的類別，並將數值變數做標準化（Normalization）供模型使用。

Num.	Attribute Name	Description	Type
1	age	The person's age in years	int
2	sex	The person's sex (1 = male, 0 = female)	Categorical
3	cp	The chest pain experienced (Value 1: typical angina, Value 2: atypical angina, Value 3: non-anginal pain, Value 4: asymptomatic)	Categorical
4	trestbps	The person's resting blood pressure (mm Hg on admission to the hospital)	int
5	chol	The person's cholesterol measurement in mg/dl	int
6	fbs	The person's fasting blood sugar (> 120 mg/dl, 1 = true; 0 = false)	Categorical
7	restecg	Resting electrocardiographic measurement (0 = normal, 1 = having ST-T wave abnormality, 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria)	Categorical
8	thalach	The person's maximum heart rate achieved	int
9	exang	Exercise induced angina (1 = yes; 0 = no)	Categorical
10	oldpeak	ST depression induced by exercise relative to rest	numeric
11	slope	the slope of the peak exercise ST segment (Value 1: upsloping, Value 2: flat, Value 3: downsloping)	Categorical
12	ca	The number of major vessels (0-3)	int
13	thal	A blood disorder called thalassemia (3 = normal; 6 = fixed defect; 7 = reversable defect)	Categorical
14	target	Heart disease (0 = no, 1 = yes)	Categorical

表 2、資料欄位特徵敘述表

2. 資料視覺化

此部分主要在於理解變數間關係，提供實務上的理解以及建立模型的特徵選取。其中資料共有 303 筆，其圖 2 顯示確診心臟病 165 人、無心臟病有 138 人，資料中心臟病確診者比例為 54.46%。此外，建立各特徵變數與目標變數 Target 之相關矩陣熱力圖，如下圖 3 所示，可以發現相關矩陣中，目標變數與其他自變數並無超過 $|r|=0.5$ ，表示無觀察到特定變數有決定性的影響因素，但仍可從中挑選相關性較大的變數依序為：exang、cp、thalach，其中亦可發現變數：chol、fbs 與目標變數之相關性較大，因此後續將針對此兩變數進行深入之視覺化分析，以決定是否將此兩遍變數納入預測模型中。

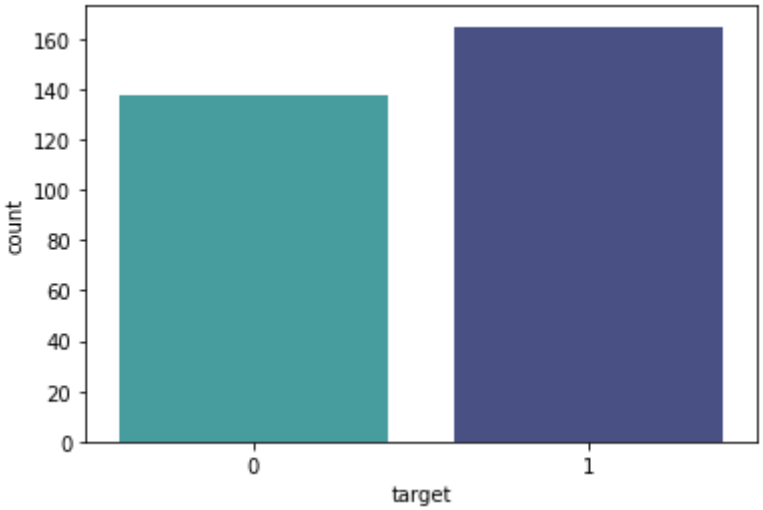

圖 2、心臟病確診分布

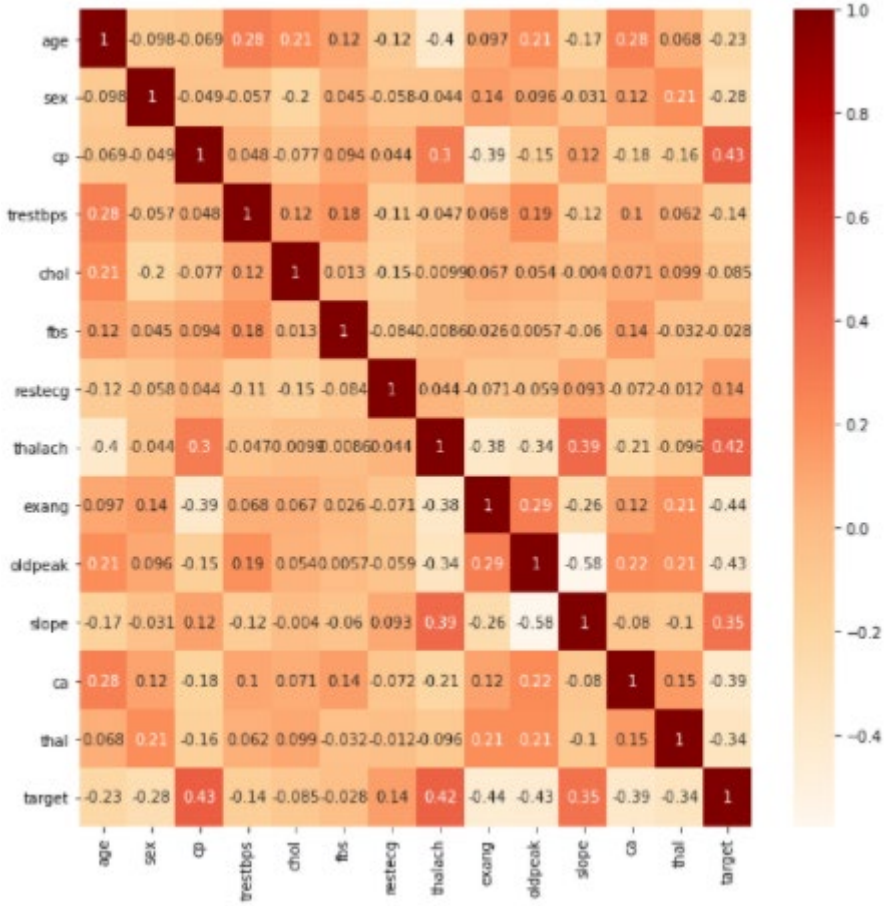

圖 3、資料集熱力圖

進一步探索資料，從特徵變數中挑選幾個與目標變數間相關之變數進行視覺化，首先將性別與目標函數進行比對，於下圖 4 中可得發現女性確診的人數雖然較少，但比例卻明顯高於男性。而於圖 5 中可得年齡也確實會影響心臟病確診的機率，但意外的是年齡越輕，其確診的比例越高，此外亦可從圖中發現最高心跳數與年齡略為負相關，但其最高心跳數的變數（thalach）將心臟病是否確診的資料拆成上下兩部分，故猜測其可能具有較強的分類能力。

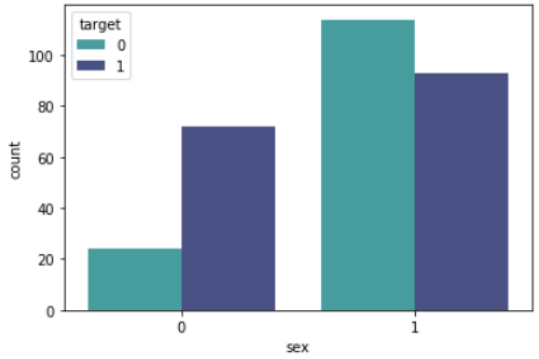

圖 4、性別與心臟病確診之分佈

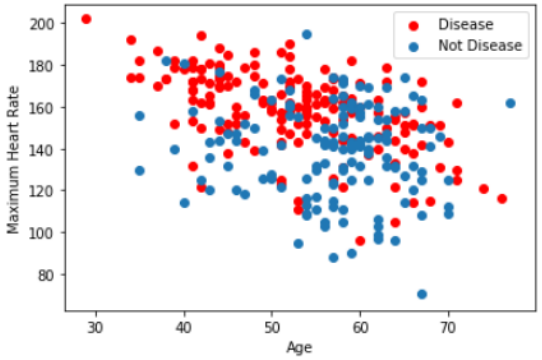

圖 5、年齡和最高心跳與心臟病確診之分佈

而其中與目標變數相關性最高的 cp、exang，兩者皆可觀察到會成為影響確診的關鍵變數，其在 cp（胸痛）中，例如： Typical Angina （典型心絞痛）的手冊者有較高比例並非心臟病。而在 exang 中，則是 exang = 1 （具 Exercise Induced Angina）時，會有較高比例並非心臟病。

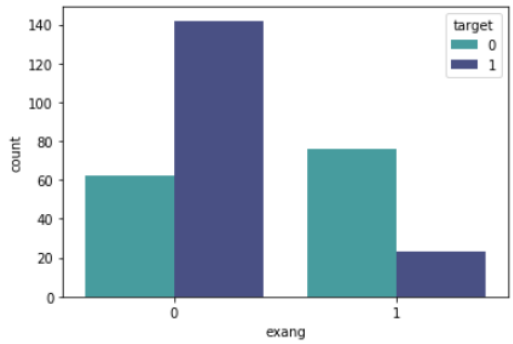

圖 6、exang 與心臟病確診之分佈

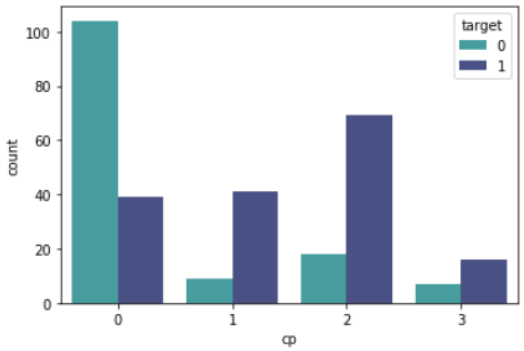

圖 7、胸痛與心臟病確診之分佈

接著分析與目標變數相關性較低的變數 chol、fbs，其中 chol (膽固醇) 數值高低並沒有將是否確診資料進行明顯區分，可以顯示其與目標函數間並沒有顯著相關，因此後續預測模型中不將其納入模型。而在 fbs（空腹血糖數值）中，不論數值是否大於 120，心臟病確診者比例並沒有太大之差異，因此亦顯示其與目標函數間並沒有顯著相關，亦不將 fbs 放入預測模型，以降低 Overfitting 發生可能性。

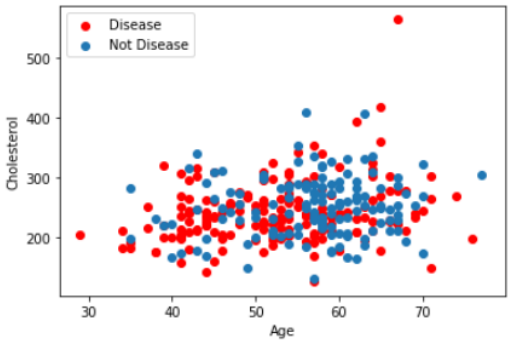

圖 7、chol 與心臟病確診之分佈

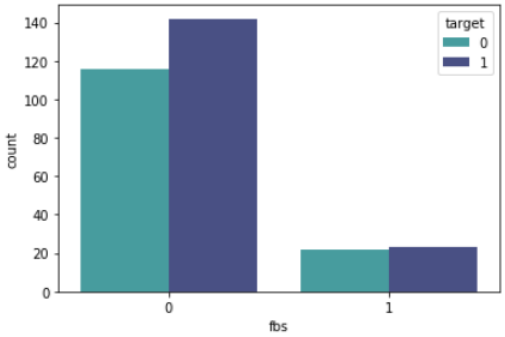

圖 8、fbs 心臟病確診之分佈

三、 資料前處理

1. 類別型變數資料轉換

為了使類別型變數於資料標準化後具有代表性，將類別型變數如：cp、thal、slope 等轉換為 Dummy Variable，例如 cp 中共分為 4 個不同類型，則新增 4 個欄位分別代表不同類型，其數值則為二分類 (0 跟 1)，以表示該筆資料屬於何種胸痛。

```
#將各類別轉換為 Dummy Variables
a = pd.get_dummies(df['cp'], prefix = "cp")
b = pd.get_dummies(df['thal'], prefix = "thal")
c = pd.get_dummies(df['slope'], prefix = "slope")
frames = [df, a, b, c]
df = pd.concat(frames, axis = 1)
df.head()
```

2. 資料特徵篩選

經資料探索中相關性與視覺化觀察後，將 chol 與 fbs 兩項與目標變數相關性較低之變數剃除，以降低 Overfitting 發生可能性，此外，由於上一步驟以將類別型變數轉換為各不同變數，因此接續將原始類別型變數欄位刪除，以避免重複將變數納入模型，特徵篩選完之資料長相如下圖 9 所示。

```
df = df.drop(columns = ['cp','thal','slope','fbs','chol'])
df.head()
```

	age	sex	trestbps	restecg	thalach	exang	oldpeak	ca	target	cp_0	cp_1	cp_2	cp_3	thal_0	thal_1	thal_2	thal_3	slope_0	slope_1	slope_2
0	63	1	145	0	150	0	2.3	0	1	0	0	0	1	0	1	0	0	1	0	0
1	37	1	130	1	187	0	3.5	0	1	0	0	1	0	0	0	1	0	1	0	0
2	41	0	130	0	172	0	1.4	0	1	0	1	0	0	0	0	1	0	0	0	1
3	56	1	120	1	178	0	0.8	0	1	0	1	0	0	0	0	1	0	0	0	1
4	57	0	120	1	163	1	0.6	0	1	1	0	0	0	0	0	1	0	0	0	1

圖 9、特徵篩選後之資料集

3. 資料標準化與切割

經過特徵篩選後，總共納入 19 項特徵變數和 1 項目標變數，其中利用 Python 語法發現各變數資料皆無任何遺失值，並透過 Summary( ) 函式以及檢視各變數分佈圖後，並無特定變數具有離群值（Outliers），故僅調整資料型態至其正確的類別，並將數值變數做標準化（Normalization）供模型使用，如下圖 10 所示。此外，本研究將資料隨機切割為訓練集(Trainging)、測試集(Testing Data)，切割比例為 7：3，Training Data: 212 筆，用以訓練並建立模型，並進行模型選擇；而 Testing Data: 91 筆，用以驗證準確度及模型效度。

	age	sex	trestbps	restecg	thalach	exang	oldpeak	ca	cp_0	cp_1	cp_2	cp_3	thal_0	thal_1	tl
0	0.950624	0.679881	0.762694	-1.004171	0.015417	-0.69548	1.085542	-0.713249	-0.943822	-0.443820	-0.633600	3.483351	-0.081379	3.972541	-1.09
1	-1.912150	0.679881	-0.092585	0.897478	1.630774	-0.69548	2.119067	-0.713249	-0.943822	-0.443820	1.573075	-0.286132	-0.081379	-0.250897	0.90
2	-1.471723	-1.465992	-0.092585	-1.004171	0.975900	-0.69548	0.310399	-0.713249	-0.943822	2.245729	-0.633600	-0.286132	-0.081379	-0.250897	0.90
3	0.179877	0.679881	-0.662770	0.897478	1.237849	-0.69548	-0.206364	-0.713249	-0.943822	2.245729	-0.633600	-0.286132	-0.081379	-0.250897	0.90
4	0.289984	-1.465992	-0.662770	0.897478	0.582975	1.43311	-0.378618	-0.713249	1.056025	-0.443820	-0.633600	-0.286132	-0.081379	-0.250897	0.90

圖 10、資料標準化之資料集

四、 模型建立

1. 神經網路預測模型

```
model_1 = Sequential()
model_1.add(Dense(32, input_dim=19, kernel_initializer='normal', activation='relu'))
model_1.add(Dropout(0.25))
model_1.add(Dense(16, kernel_initializer='normal', activation='relu'))
model_1.add(Dropout(0.25))
model_1.add(Dense(2, activation='softmax'))

# compile model
Adam(lr=0.01)
model_1.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model_1.summary()
```

```
history_1=model_1.fit(x_train, y_train, validation_data=(x_test, y_test),epochs=50, batch_size=10)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_210 (Dense)            (None, 32)                640
_________________________________________________________________
dropout_136 (Dropout)        (None, 32)                0
_________________________________________________________________
dense_211 (Dense)            (None, 16)                528
_________________________________________________________________
dropout_137 (Dropout)        (None, 16)                0
_________________________________________________________________
dense_212 (Dense)            (None, 2)                 34
=================================================================
Total params: 1,202
Trainable params: 1,202
Non-trainable params: 0
_________________________________________________________________
```

圖 11、模型架構及參數圖

利用 python keras 初始建構深度學習預測模型，模型之架構及參數如下圖 11，總共架設 3 個 dense 層，其中第一層 dense 為 32 個節點，而第二層 dense 為 16 個節點，前兩層之 Activation Function 設定為 relu，以有效克服梯度消失之問題，最後輸出層使用 Activation Function 設定為 softmax，以判斷是否確診心臟病之分類問題，其中未必免 Overfitting，於各層級間加入 Dropout 以拿走部分神經元，使神經元權重變化更不敏感，此外，由於目標函數為 0 與 1 之整數判斷是否確診，所以選用 sparse_categorical_crossentropy 之 Loss Function，更合理檢視該模型效度，而相關超參數設定如下：kernel_initializer = normal、optimizer = adam、learning rate=0.01、epoch=50、batch size = 10。

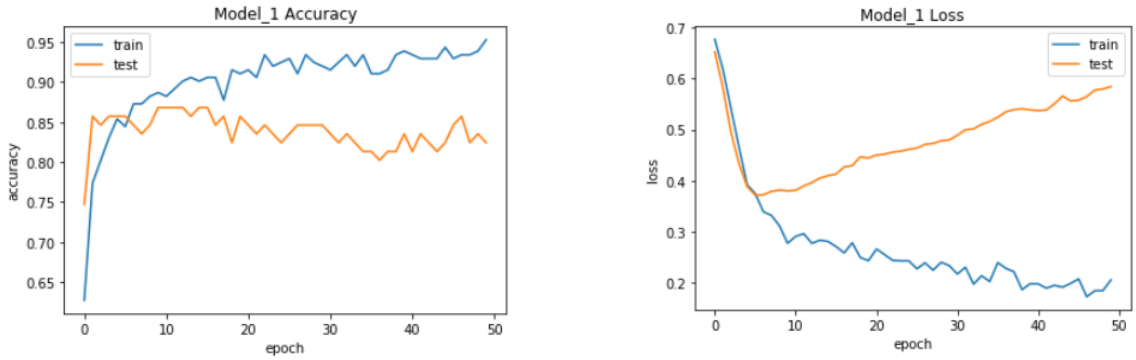

圖 12、Training & Testing 之 Accuracy & Loss

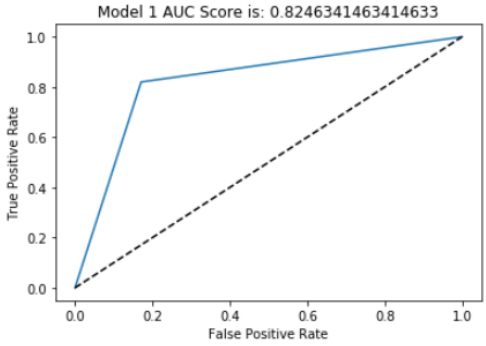

圖 13、預測模型之 ROC & AUC

此神經網路預測模型 testing 準確率為 82.42% 與 AUC 為 0.8246，其中以 AUC (Area under curve)為評透過 ROC Curve 所建構出的面積，此為普遍進行二元分類(Classification)的成效分析方法，其值介於[0,1]，數值越接近 1 則表現越好。從圖 12 可以發現當 epoch 越大時，Training Accuracy 有越來越大的趨勢，但 Testing Accuracy 卻沒有隨之增長，加上 epoch 在大於 10 時，loss 不減反增，此推斷為出現 Overfitting 過擬合之問題，因此於下一部分進行超參數調整，以針對過擬合進行處理與提升模型準確度與 AUC。

2. 超參數調整

I. 增加 kernel Regularizer

為了解決上述之問題，於模型前後各加入了 L2 kernel regularizer 防止模型過擬合，其中 L2 kernel regularizer 設定為 0.01，得以讓神經元權重正規化，於權重矩陣加入懲罰性函數，以有效防止過擬合問題，其結果如下圖 14、15 所示，神經網路預測模型 testing 準確率上升為 84.62%，AUC 為 0.8468，雖然 loss 上升幅度經改善後有逐漸趨緩，不過仍有改善空間，因此持續進行優化。

```
#增加kernal regularizer
model_2 = Sequential()
model_2.add(Dense(32, input_dim=19, kernel_initializer='normal',kernel_regularizer=regularizers.l2(0.01), activation='relu'))
model_2.add(Dropout(0.25))
model_2.add(Dense(16, kernel_initializer='normal',kernel_regularizer=regularizers.l2(0.01), activation='relu'))
model_2.add(Dropout(0.25))
model_2.add(Dense(2, activation='softmax'))

# compile model
Adam(lr=0.01)
model_2.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model_2.summary()
```

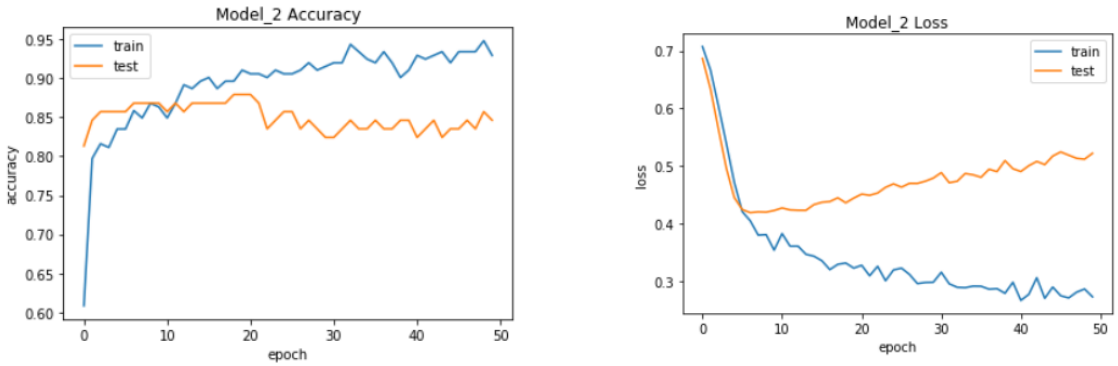

圖 14、Training & Testing 之 Accuracy & Loss

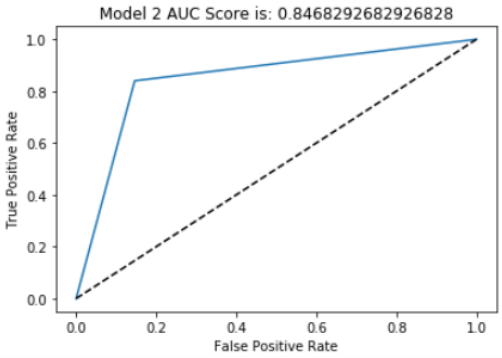

圖 15、預測模型之 ROC & AUC

II. Activation Function 更改為 sigmoid

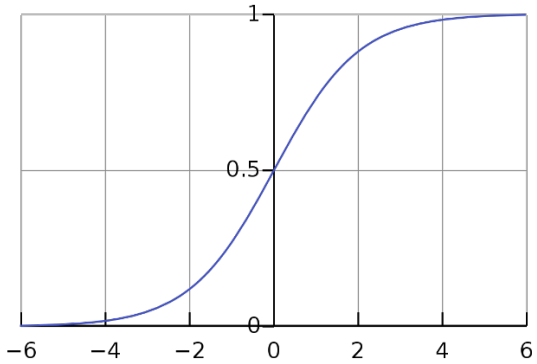

圖 16、sigmoid 數學函式圖

原先 Activation Function 設定為 softmax，但其多運用於多分類問題，然而本研究專注於二分類問題，因此將輸出層之 Activation Function 更改為 sigmoid，其數學函式如上圖 16 所示，經由此修正讓模型可以更有效進行分類，模型架構如下圖 17 所示，其結果如下圖 18、19 所示，該預測模型 testing 準確率上升為 85.71%，AUC 為 0.859，loss 上升幅度更為趨緩，接著持續進行過擬合修正。

```
# Activation Function 更改為 sigmoid，適合用於二分類
model_3 = Sequential()
model_3.add(Dense(32, input_dim=19, kernel_initializer='normal',kernel_regularizer=regularizers.l2(0.01), activation='relu'))
model_3.add(Dropout(0.25))
model_3.add(Dense(16, kernel_initializer='normal',kernel_regularizer=regularizers.l2(0.01), activation='relu'))
model_3.add(Dropout(0.25))
model_3.add(Dense(2, activation='sigmoid'))

# compile model
Adam(lr=0.01)
model_3.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model_3.summary()
```

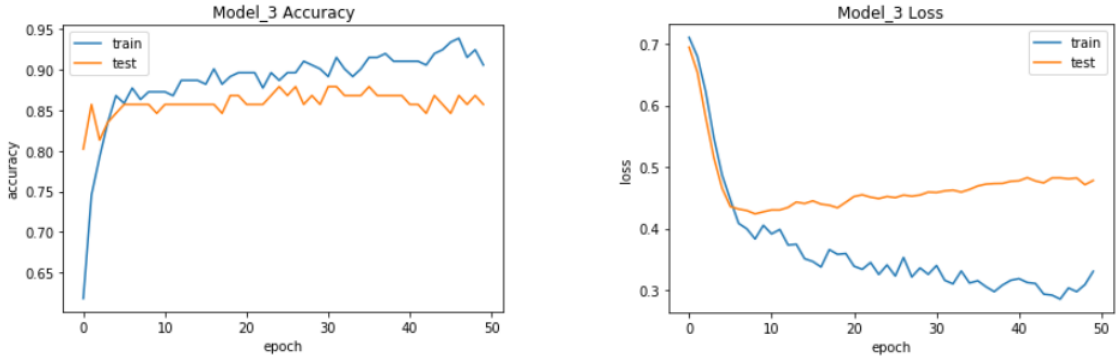

圖 18、Training & Testing 之 Accuracy & Loss

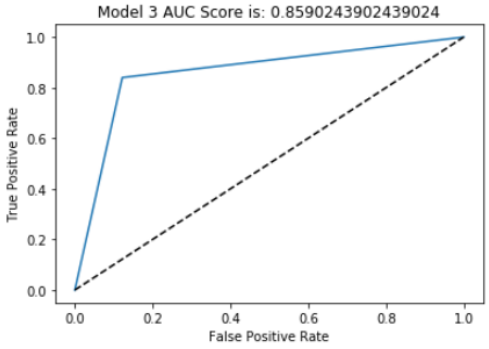

圖 19、預測模型之 ROC & AUC

III. Epochs 調整為 10

經由上述三個預測模型，皆可發現當 epochs 大於 10 時，loss 都會隨之上升，因此為直接防止過擬合，將 epochs 下修為 10，讓模型得以正常訓練，其結果如下圖 20、21 所示，神經網路預測模型 testing 準確率上升為 86.81%，AUC 為 0.8690，準確率相較於第一個建立之模型，已有顯著的改善，接續針對其他超參數進行調整。

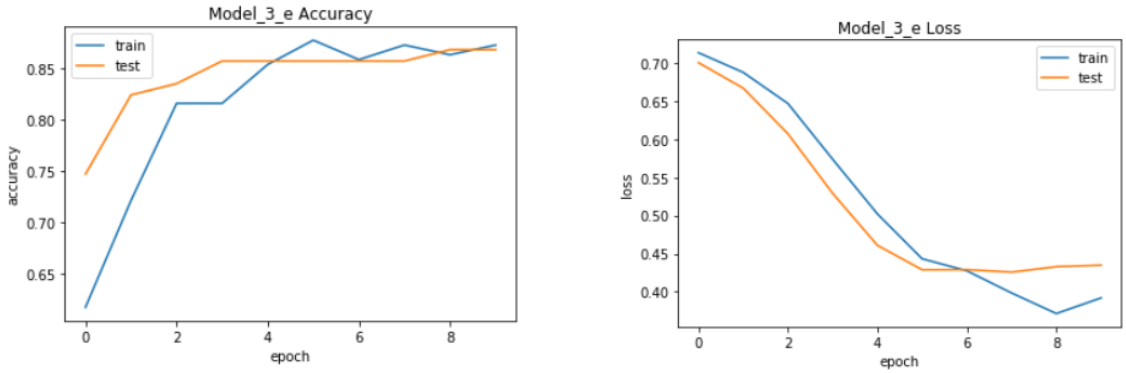

圖 20、Training & Testing 之 Accuracy & Loss

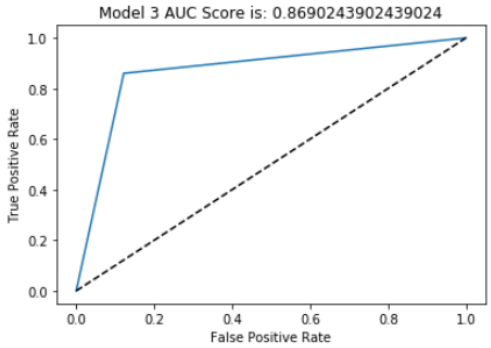

圖 21、預測模型之 ROC & AUC

IV. Learning Rate & Batch Size

針對 Learning Rate 進行調整，本研究比較三種 Learning Rate：0.1、0.01、0.001，下表 3 為分析比較之結果，其中可以發現 Learning Rate =0.1、0.01 時，除在收斂速度上有差別外，對於學習成效沒有太大的影響，可能是因為此 model 的參數眾多，使得 local minimum 較不可能出現，而當 Learning Rate=0.001 時，Testing Accuracy 與 AUC 成效表現較差，推斷可能為因為學習率過低，導致 Overfitting 之問題產生，因此保留 Learning Rate=0.01。

表 3、Learning Rate 評估比較表

	Training	Testing	AUC
0.1	0.8820	0.8681	0.8690
0.01	0.8773	0.8681	0.8690
0.001	0.8762	0.8461	0.8468

接著針對 Batch Size 進行調整，本研究比較三種 Batch Size：5、10、15，下表 4 為分析比較之結果，其可以發現當 Batch Size =15 時結果相對較差，而當 Batch Size =5 與 10 時，結果沒有任何差異，因此持續選用 Batch Size=10，使學習模型可以更加穩定。

表 4、Learning Rate 評估比較表

	Training	Testing	AUC
5	0.8915	0.8681	0.8690
10	0.8773	0.8681	0.8690
15	0.8679	0.8571	0.8590

V. kernel regularizer (0.01 vs. 0.001)

此外，試著調整 kernel regularizer 的參數，嘗試 0.01 和 0.001，比較結果如下表 5 所示，發現當懲罰參數調降為 0.001 時，結果相對較不好，可能是因為目標函數為 0 與 1 之變數，所以當懲罰參數越大時，有著越好的成效。

表 5、Learning Rate 評估比較表

	Training	Testing	AUC
0.01	0.8915	0.8681	0.8690
0.001	0.8773	0.8571	0.8590

3. 模型評估與比較

經由上述超參數調整之結果，與最初模型相比，增加了 kernel Regularizer、Activation Function 更改為 sigmoid 與 Epochs 調整為 10，而 learning rate、batch size 與 kernel regularizer 分別為 0.01、10 與 0.01，testing 準確率 86.81%，AUC 為 0.8690。

接著為了驗證此模行之效度，使用機器學習方法建立羅吉斯歸方法 Logistic regression 及隨機森林分類器以預測模型，模型架構及程式如圖 22、圖 23，兩者模型經訓練後之 Testing 準確率分別為 83.52% 及 82.42%，而 AUC 為 0.8357 與 0.8246，可見兩者效度與最終模型相差甚遠，由下表 5 可見，說明本研究建構的深度學習模型在此資料集中有最好的表現。

```
# Sklearn Logistic Regression
lr = LogisticRegression()
lr.fit(x_train,y_train)
acc_lr_train = lr.score(x_train,y_train)
acc_lr_test = lr.score(x_test,y_test)
```

圖 22、Logistic Regression 模型架構

```
# Random Forest
rdf=RandomForestClassifier(n_estimators=10,criterion='entropy',random_state=38)
rdf.fit(x_train,y_train)
acc_rdf_train = rdf.score(x_train,y_train)
acc_rdf_test = rdf.score(x_test,y_test)
```

圖 23、Random Forest 模型架構

表 6、各訓練模型評估比較

訓練模型	Testing Acc	AUC
Logistic Regression	0.8351	0.8357
Random Forest	0.8241	0.8246
深度學習預測模型	0.8681	0.8690

五、 成果與結論

1. 研究結果

總結以上，經由資料前處理(包含資料相關分析與資料視覺化)等手法找出特徵值作為訓練模型變數，並配合資料轉換與標準化，最終挑選出 19 項變數作為訓練模型之因子，而透過 keras 深度學習模型建立，並透過實驗針對 activation function、epochs、Adam learning rate、kernel regularizer 等超參數進行優化，最終得出之訓練模型與其他機器學習相比有著最好的表現。

2. 未來發展

從本次的 Project 中，可以發現適當的篩選特徵變數及資料前處理手法對於後續的建立 Model 及辨識準確率極為重要，其中初始模型建立時遇到過擬合之問題，而對於深度學習之各項超參數的理解後，逐漸理解如何適當調整參數，使模型帶來更好的表現，也是報告帶給我的學習重點，所以我認為

後續可以持續建立不同深度學習模型並依資料處理後之特性來調整超參數或是運用於其他精準醫療預測資料集(例如，皮膚癌、肺癌、糖尿病與急診室住院預測等)來實作驗證，以此延伸本次研究內容與實際落地。