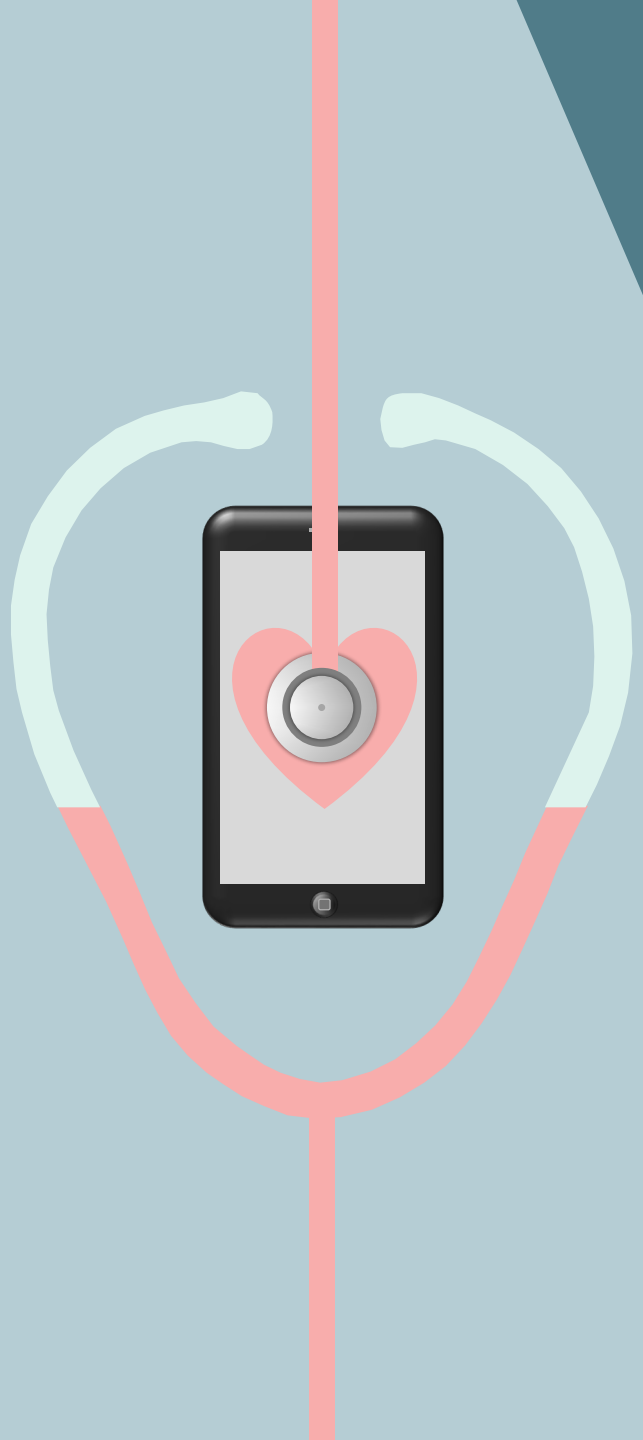
基於深度學習神經網路於心臟病確診分類之應用

指導老師：邱銘傳 教授　　組別：Group 8

學生：陳譽升 109034538



Agenda

01 研究背景介紹

02 資料探索與前處理

03 模型建立與超參數優化

04 模型評估與比較

05 結論與未來展望

背景-心臟病確診預測

於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

將個人生理資訊建立於資料庫，並透過建立模型預測來輔佐決策，以提升臨床醫療的正確性

背景-5W1H

Power by
資料視覺化、機器學習、深度學習

WHY
在成本控制下提供更快速與精準之確診判斷，
使病患得以安心接受後續相關治療

WHAT
心臟病確診預測

WHO
心臟不適之民眾與心臟病科醫生

WHEN
欲了解自身是否罹患心臟病時

WHERE
醫院及診所

心臟病預測模型研究流程

資料探索

1. 資料集讀取與解析
2. 資料相關性分析
3. 資料視覺化

資料前處理

4. 類別型資料轉換
5. 特徵篩選
6. 資料標準化
7. 資料切割

模型建立

8. NN 模型建立
9. 超參數調整
10. 模型評估與比較

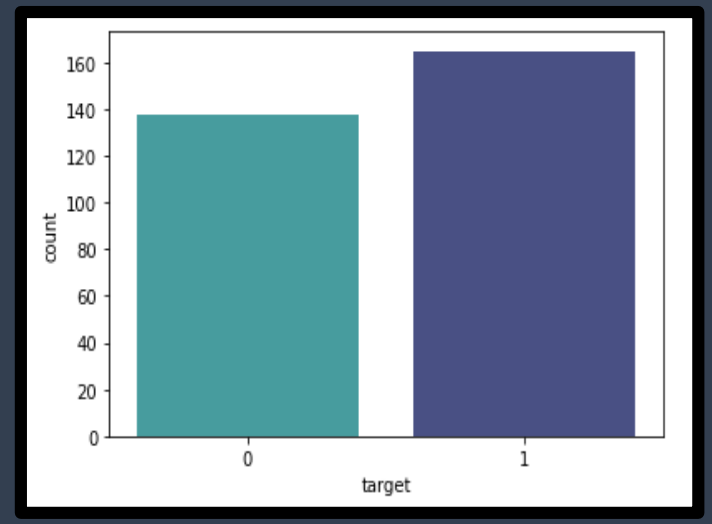
資料集說明

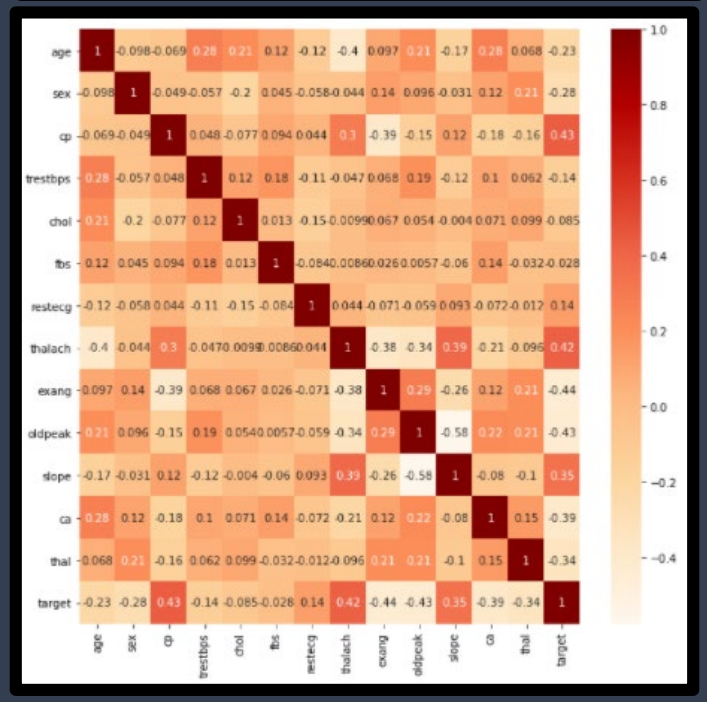
1. 使用 UCI Machine Learning Repository 心臟病預測資料庫
2. 數據共有303筆，14項變數
3. 目標應變數為：1為有心臟病、0為無心臟病
4. 右表為其資料表類別與屬性表
5. 經過 Python讀取資料後，發現並無任何遺失值
6. 透過 Summary( ) 函式以及檢視各變數分佈圖後，無特定變數具有離群值

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

資料視覺化

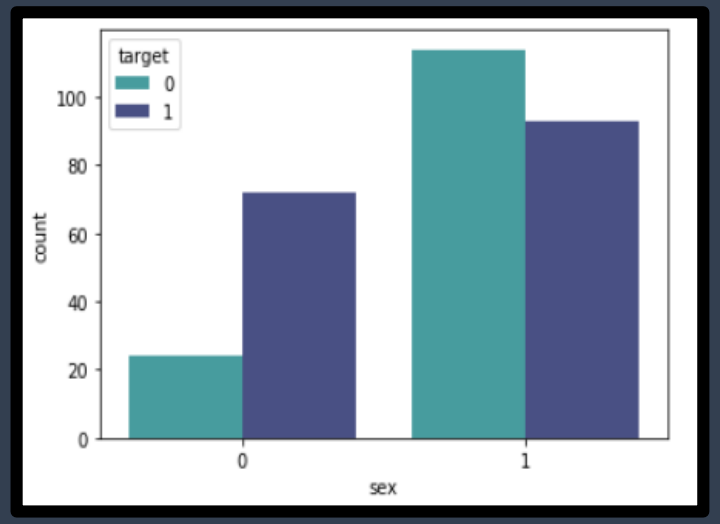
1. 主要在於理解變數間關係，提供實務上的理解以及建立模型的特徵選取
2. 確診心臟病有165人、無心臟病有138人
3. 確診者比例占總體之54.46%
4. 從中挑選相關性較大的變數依序為：exang、cp、thalach
5. 亦可發現變數：chol、fbs與目標變數之相關性較大

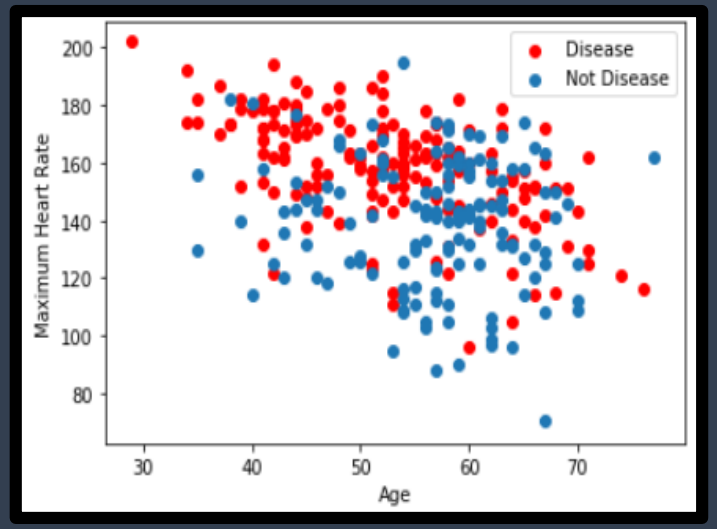




探索資料

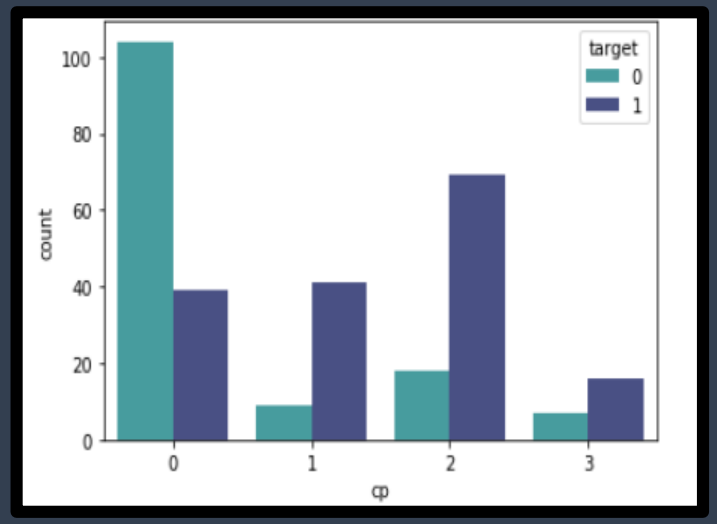
1. 將性別與目標函數進行比對，可發現女性確診的人數雖然較少，但比例卻明顯高於男性
2. 年齡也確實會影響心臟病確診的機率，年齡越輕，其確診的比例越高
3. 發現最高心跳數與年齡略為負相關，但其最高心跳數的變數（thalach）將心臟病是否確診的資料拆成上下兩部分，故猜測其可能具有較強的分類能力

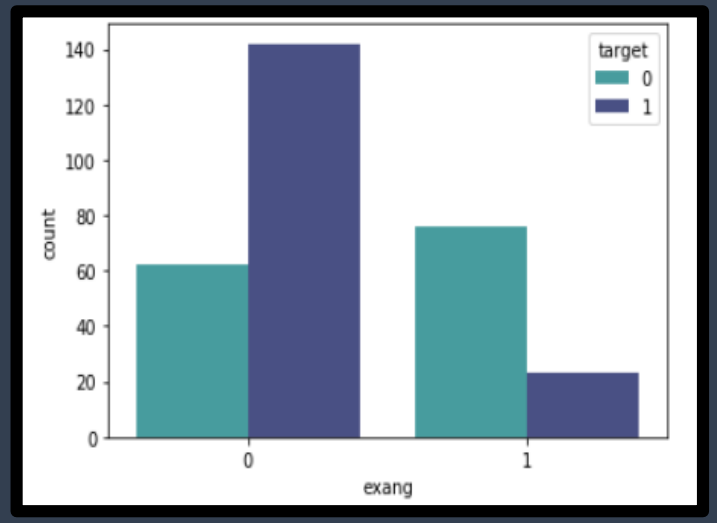




探索資料 (Cont.)

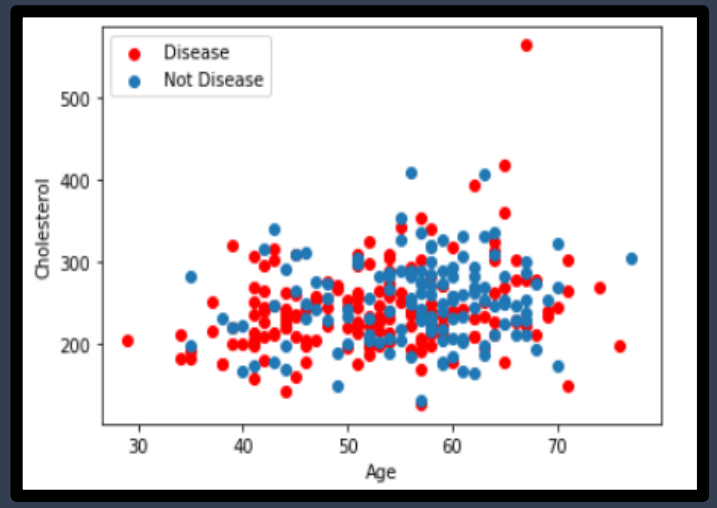
3. 與目標變數相關性最高的 cp、exang，其兩者皆可觀察到會成為影響確診的關鍵變數，其在 cp（胸痛）中
4. 在 exang 中則是 exang = 1（具 Exercise Induced Angina）時，有較高比例並非心臟病

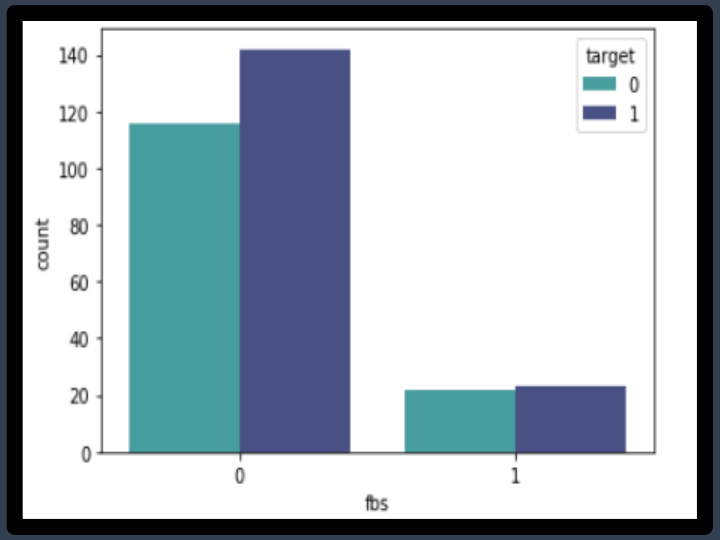




探索資料 (Cont.)

5. 在chol (膽固醇) 數值高低並無將是否確診資料進行明顯區分，故後續預測模型中不將其納入模型
6. 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
7. 透過上述方法降低Overfitting發生可能性





資料前處理 step 1 & 2

類別型變數資料轉換

為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

```
#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()
```

資料特徵篩選

將原始類別型變數欄位與目標函數相關性較小之變數：fbs、chol 刪除

```
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							
3	56	1	120	1	178	0	0.8	0	1	0	1	0	0							
4	57	0	120	1	163	1	0.6	0	1	1	0	0	0							

19項特徵變數

1項目標變數

資料前處理 step 3 & 4

資料標準化與切割

將資料進行標準化處理，以供模型使用

隨機切割為訓練集(Training)、測試集(Testing Data)，切割比例為7：3

	age	sex	trestbps	restecg	thalach	exang	oldpeak	ca	cp_0	cp_1	cp_2	cp_3	thal_0	thal_1	th
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

神經網路預測模型 Original Model

```
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)
```

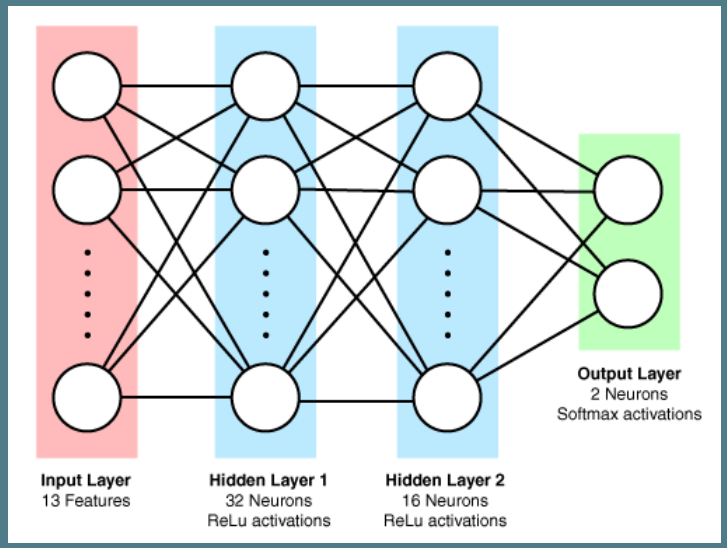
Activation Function

- Hidden Layer：ReLu
- Output Layer：Softmax

Loss Function

- sparse_categorical_crossentropy

```
_________________________________________________________________
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
_________________________________________________________________
```



kernel_initializer

- normal

Learning Rate

- Adam / 0.01

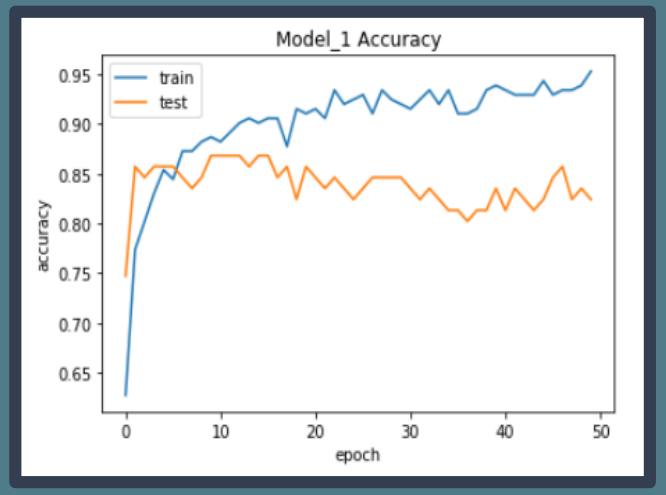
Epochs

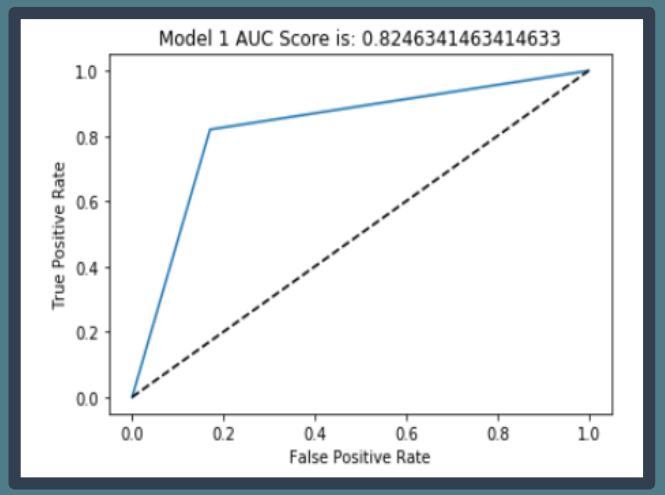
- 50

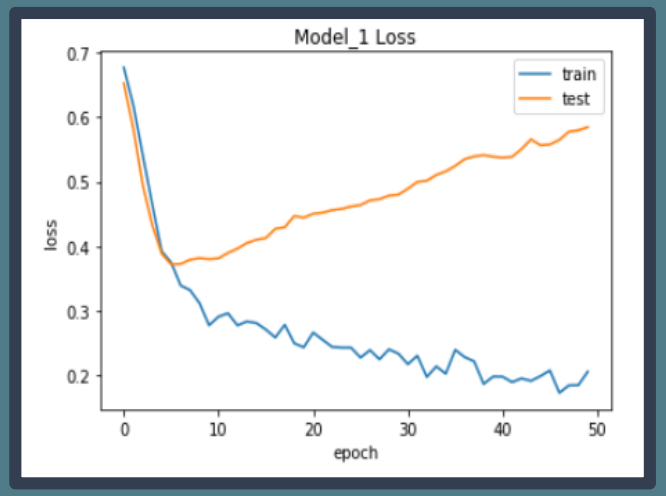
Batch Size

-10

神經網路預測模型 Original Model







Testing Acc	/	AUC
82.42%	/	0.8246

Overfitting

↓

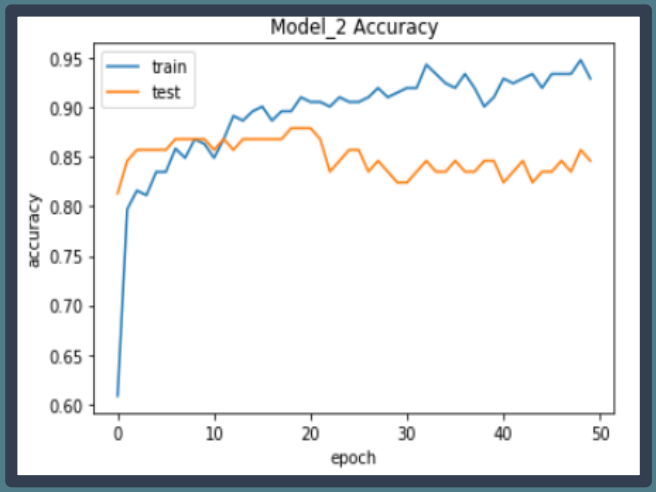
超參數優化

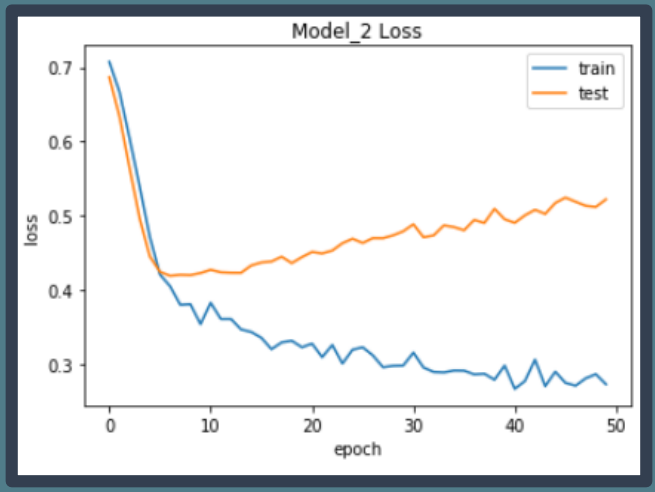
超參數調整 by增加kernel Regularizer

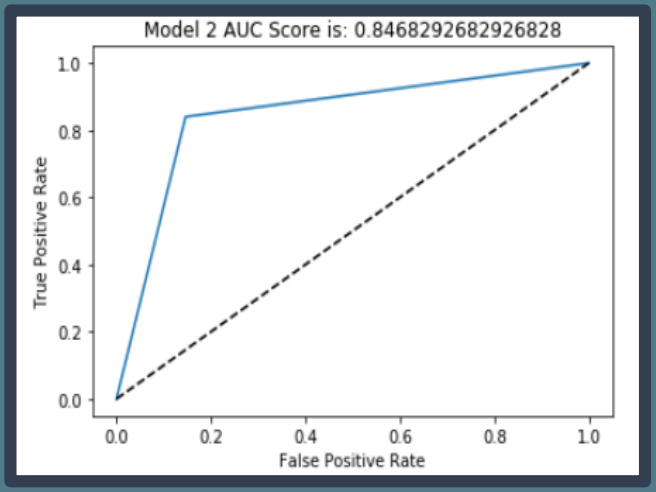
```
#增加kernal regularizer
model_2 = Sequential()
model_2.add(Dense(32, input_dim=19, kernel_initializer='normal',kernel_regularizer=regularizers.l2(0.01), activation='relu'))
model_2.add(Dropout(0.25))
model_2.add(Dense(16, kernel_initializer='normal',kernel_regularizer=regularizers.l2(0.01), activation='relu'))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

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，以防止過擬合問題







Testing Acc
- 84.62%

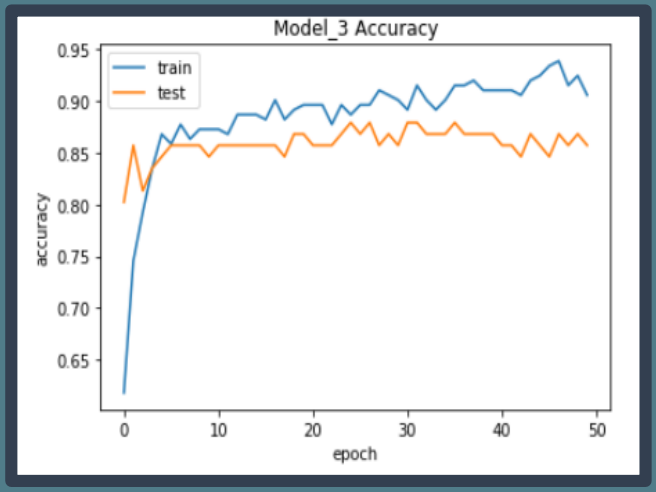
AUC
- 0.8468

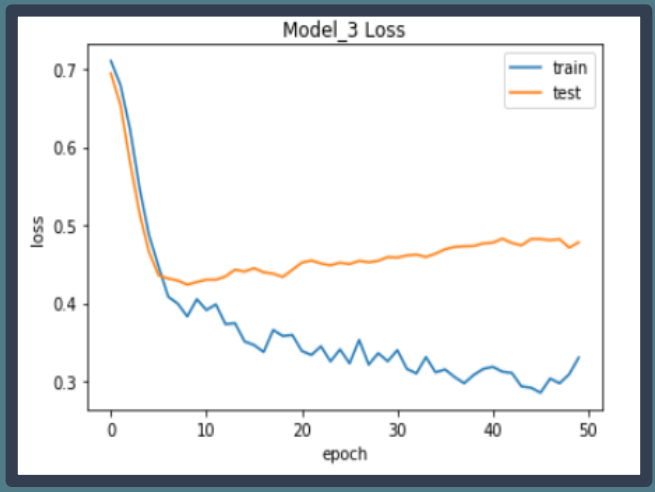
超參數調整 by 調整 Activation = sigmoid

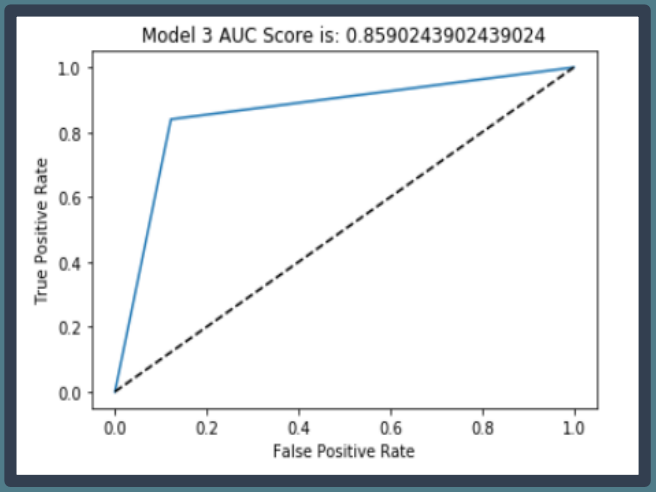
```
# Activation Function 更改為 sigmoid，適合用於二分類
model_3 = Sequential()
model_3.add(Dense(32, input_dim=19, kernel_initializer='normal',kernel_regularizer=regularizers.l2(0.01), activation='relu'))
model_3.add(Dropout(0.25))
model_3.add(Dense(16, kernel_initializer='normal',kernel_regularizer=regularizers.l2(0.01),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

將輸出層之 Activation Function 更改為sigmoid，更有效進行二分類預測問題



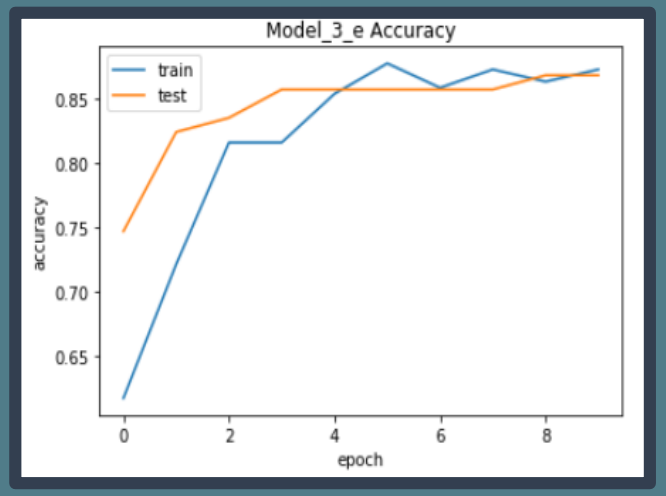


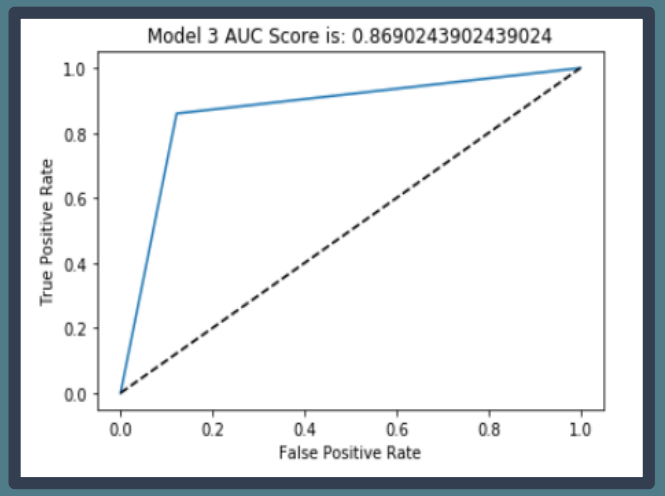


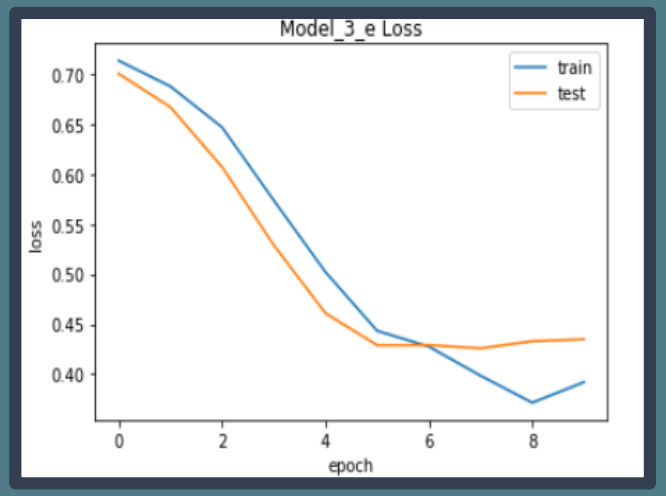
Testing Acc
- 85.71%

AUC
- 0.8590

超參數調整 by 調整 Epochs = 10







Testing Acc	/	AUC
86.81%	/	0.8690

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

超參數調整 by 調整其他超參數

Learning Rate (0.1/0.01/0.001)

除在收斂速度上有差別外，對於學習成效沒有太大的影響，
當Learning Rate=0.001時，Testing Accuracy與AUC成效表現較差

	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

Batch Size (5/10/15)

當Batch Size =15時結果相對較差，
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

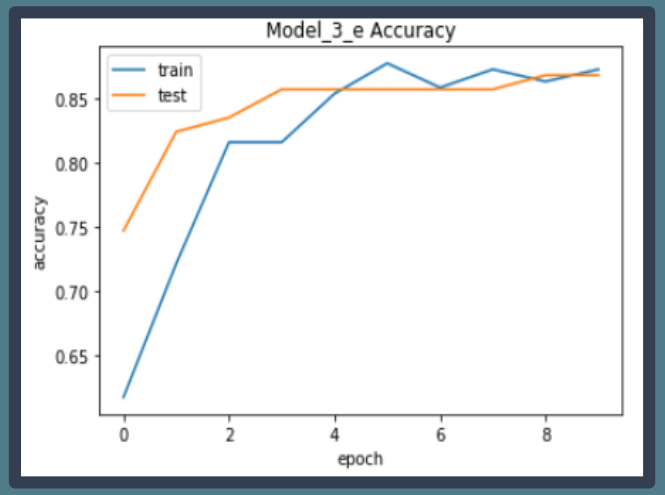
	Training	Testing	AUC
5	0.8915	0.8681	0.8690
10	0.8773	0.8681	0.8690
15	0.8679	0.8571	0.8590

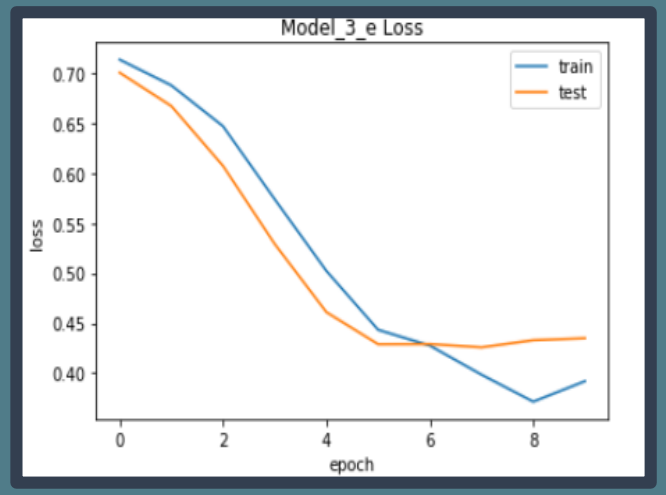
模型評估與比較 Final Model Evaluation

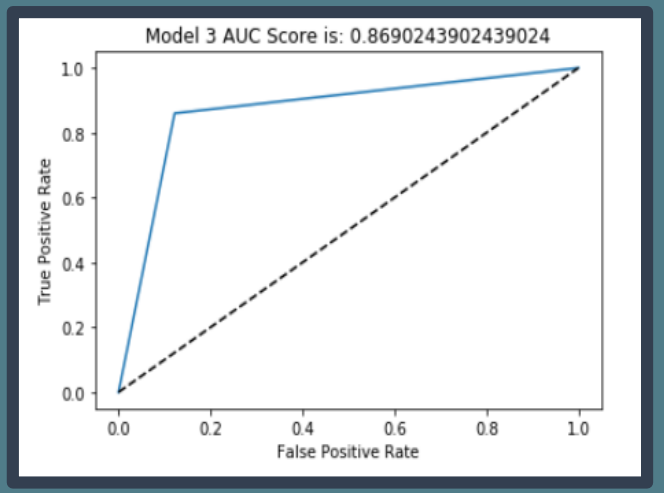
Final Model

1. 增加 Kernel Regularizer
2. Activation = sigmoid
3. Epochs = 10
4. Learning Rate = 0.01
5. Batch Size = 10
6. Kernel Regularizer = 0.01

Testing Accuracy = 86.81%
AUC = 0.8690







模型評估與比較 by compare with ML models

建立機器學習演算法
-Logistic Regression
-Random Forest
以評估深度學習模型之效度

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

```
# 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)
```

```
# 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)
```

最終深度學習模型
有著最佳預測效果

結論與未來展望

年齡

性別

⋮

最高心跳數

血氧濃度

心臟病確診預測
深度學習模型

罹患心臟病

無罹患心臟病

導入實際應用現場獲取更多資料，提升模型穩定度

資料前處理與預測模型超參數不斷優化更新

精準醫療資料集預測模型建立