人工智慧模型模擬麻醉醫生以評估手術中病人的清醒程度

Artificial intelligence model simulates anesthesiologist to assess the awareness of patients during surgery

指導教授： 謝建興 教授
學 生： 黃偉豪
實 驗 室： 智慧型控制實驗室

摘要

國外已不少有患者在手術中甦醒的案例，患者清楚感覺到執刀醫生正在對自己身體進行手術，因此讓病患留下永遠的心理傷害。腦波雙頻指數(BIS)是其中一種清醒程度評估方法，讓執刀醫生了解病人手術中的清醒狀態，但是腦波雙頻指數(BIS)判斷的精準度相較於麻醉醫生的真人評估仍略遜一些，此外，目前麻醉醫生的真人評估是在術後進行，無法在手術中提供執刀醫生準確的病人麻醉清醒程度。

因此，我們將台大醫院 2014 年 15 場手術資料當作實驗資料，並且使用 Jupter Notebook 軟體中 Python 的 scikit-learn 機器學習函式庫與其他函式庫等等，使用多種機器學習演算法模擬麻醉醫生以評估手術中病人的清醒程度，在手術中提供執刀醫生更準確的病人麻醉清醒程度。本研究使用多階線性回歸、調整超參數後的決策樹回歸和調整超參數後的隨機森林回歸演算法來模擬三位麻醉醫生並對測試資料進行預測評估，並比較各種回歸模型的決定係數與預測誤差，最終選擇最佳的回歸模型並加以保存，即可在手術中直接預測病人手術中的清醒程度，優化原本臨床上其中一種清醒程度評估方法(腦波雙頻指數(BIS))，提供執刀醫生更準確的病人麻醉清醒程度，避免病人在手術中甦醒且造成病人永遠的心理傷害。

關鍵詞: 麻醉清醒程度、腦波雙頻指數(BIS)、多階線性回歸、決策樹回歸、隨機森林回歸

Abstract

There have been many cases in which patients had awakened during surgery. The patients clearly felt that the surgeon was performing surgery on their body, thus leaving the patient with permanent psychological harm. The Bispectral Index (BIS) is one of the awake assessment methods that allows the surgeon to understand the awareness of the patient during surgery. But the accuracy of the Bispectral index (BIS) compared with the anesthesiologist's assessment is still slightly worse. In addition, the anesthesiologist's assessment is performed after the operation, and it is impossible to provide the doctor with patient’s accurate awareness during the operation.

Therefore, I used fifteen surgical data from National Taiwan University Hospital as experimental data, and used Python's machine learning library “scikit-learn” and other

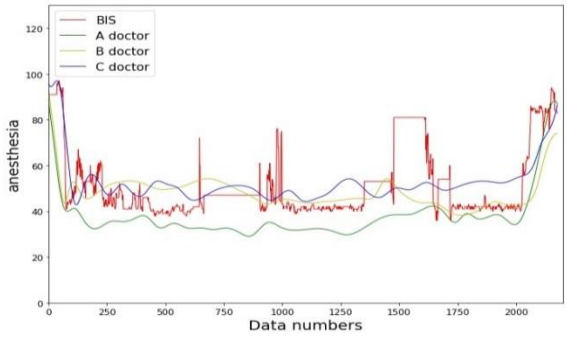
libraries in the Jupyter Notebook software. I used multiple machine learning algorithms to simulate multiple anesthesiologists to assess the awareness of patients during surgery in order to replace the existing assessment method in National Taiwan University Hospital. In this study, multi-order linear regression, tree and random forest algorithm after adjusting hyperparameters were used to simulate three anesthesiologists to predict the data in test set. Comparing with the coefficient of determination(R^2 or r^2) and the error of prediction of each regression models, and finally select the best regression model and save it, it can directly predict the awareness of the patient during surgery. It optimizes one of the clinically conscious evaluation methods (BIS), and it provides the doctor with more accurate patient' consciousness to avoid the patient from waking up during surgery and getting a permanent psychological damage.

Keywords: awareness, Bispectral index(BIS), multi-order linear regression, decision tree regression, random forest regression

二、前言

國外已不少有患者在手術中甦醒的案例，患者能清楚地感覺到執刀醫生正在對自己身體進行手術，雖然醒來的狀態通常持續的時間很短，然而卻可能讓病患留下一輩子的陰影[1] 。

在臨床上，腦波雙頻指數(BIS)讓執刀醫生了解病人手術中的清醒狀態，其數值為 0~100，越高代表患者清醒程度越高，反之，清醒程度越低。但是 BIS 判斷的精準度相較於麻醉醫生的真人評估仍略遜一些，如圖一所示[2] 。



圖一、一場手術中 BIS 值與三位醫生評估值的比較

目前最準確的評估方法為麻醉醫生根據手術中患者的生命體徵，例如:心律、動脈血壓、麻醉劑種類、藥物用量等等評估患者當下的清醒程度，但是此真人評估是在術後才進行，無法在手術中提供醫生病人的清醒程度。

本研究的實驗資料來自於 2014 年台大醫院與元智智慧型控制實驗室合作所提供的 70 場手術，本研究取其中 15 場手術當作實驗資料，並且使用 Jupter Notebook 軟體並以 Python 程式語言為基礎，安裝機器學習相關的函式庫，例如:scikit-learn 機器學習函式庫等等，導入函式庫中的多階線性回歸、決策樹回歸和隨機森林回歸模型，並且使用訓練集加以訓練，讓模型模擬麻醉醫生評估手術中病人的清醒程度，此外，透過調整超參數來優化原本的決策樹回歸和隨機森林回歸。最後，比較各種回歸模型對測試集的決定係數與預測誤差大小，選擇最佳的模型並使用 joblib 函式將其保存成 pkl 檔案，未來在手術中載入此檔案的模型，即可直接預測病人手術中的清醒程度，提供執刀醫生更準確的病人清醒程度，優化原本臨床上其中一種清醒程度評估方法(腦波雙頻指數(BIS))。

三、實驗方法

3.1 介紹原始資料集跟測試集

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Time	HR	PVC	SPO2	Pulse	NBP_Dia	NBP_Sys	NBP_Mean	ABP_Dia	ABP_Sys	ABP_Mean	RESP	BIS	SQI	EMG	SR	MAC	A_Doctor_value	B_Doctor_value	C_Doctor_value	Average_doctor_value
2	########	-1	-1	99.1	96	-1	-1	-1	--	--	--	--	--	--	--	----		87.022252	92.645537	95.66987	91.77921967
3	########	-1	-1	99.3	94	-1	-1	-1	--	--	--	--	--	--	--	----		86.33083506	91.79864095	95.39612495	91.17520032
4	########	-1	-1	99.5	93	-1	-1	-1	--	--	--	--	--	--	--	----		85.62434251	90.96002557	95.16117063	90.58184624
5	########	-1	-1	99.3	94	-1	-1	-1	--	--	--	--	--	--	--	----		84.90348141	90.12911096	94.9631453	89.99857922
6	########	-1	-1	99.4	94	-1	-1	-1	--	--	--	--	--	--	--	----		84.16895721	89.30532226	94.80018887	89.42482278
7	########	-1	-1	99	94	-1	-1	-1	--	--	--	--	--	--	--	----		83.42147771	88.48808062	94.67043926	88.8599992
8	########	-1	-1	98.7	93	-1	-1	-1	--	--	--	--	--	--	--	----		82.66174971	87.67680856	94.57203453	88.30353093
9	########	-1	-1	99.1	95	-1	-1	-1	--	--	--	--	--	--	--	----		81.89047831	86.87092892	94.50311343	87.75484022
10	########	-1	-1	99.4	98	76	137	91	--	--	--	--	--	--	--	----		81.1083716	86.06986505	94.46181436	87.21335034
11	########	-1	-1	99.9	96	76	137	91	--	--	--	--	--	--	--	----		80.31613521	85.27303868	94.4462766	86.6784835
12	########	93	-1	100	95	76	137	91	--	--	--	--	--	--	--	----		79.51447622	84.47987325	94.45463698	86.14966215
13	########	92	0	100	95	76	137	91	--	--	--	--	--	--	--	----		78.7041013	83.68979134	94.48503605	85.62630956
14	########	91	0	99.3	94	76	137	91	--	--	--	--	--	--	--	----		77.88571752	82.90221554	94.5356111	85.10784805
15	########	93	0	99	92	76	137	91	--	--	--	--	--	--	--	----		77.06003005	82.11656771	94.60450051	84.59369942
16	########	92	0	99.1	92	76	137	91	--	--	--	--	--	--	--	----		76.22774605	81.33227187	94.68984333	84.08328708
17	########	93	0	99.1	95	76	137	91	--	--	--	--	--	--	--	----		75.38957263	80.54874966	94.78977797	83.57603342
18	########	91	0	99.4	89	76	137	91	--	--	--	--	--	--	--	----		74.54621612	79.76542359	94.90244294	83.07136088
19	########	90	0	99.3	89	76	137	91	--	--	--	--	--	--	--	----		73.69838264	78.98171752	95.02597643	82.5686922
20	########	92	0	99.1	90	76	137	91	--	--	--	--	-1	0	-1	-1--		72.84677963	78.19705287	95.15851721	82.0674499
21	########	93	0	99.4	89	76	137	91	--	--	--	--	-1	0	-1	-1--		71.99211242	77.41085223	95.29820406	81.56705624
22	########	93	0	99.6	92	76	137	91	--	--	--	--	-1	0	-1	-1--		71.13508926	76.6225402	95.44317454	81.06693467
23	########	96	0	100	93	76	137	91	--	--	--	--	-1	0	-1	-1--		70.27641598	75.83153701	95.59156844	80.56650714
24	########	98	0	100	94	76	137	91	--	--	--	--	-1	0	-1	-1--		69.41679849	75.03726655	95.74152286	80.06519597
25	########	95	0	100	95	76	137	91	--	--	--	--	-1	0	-1	-1--		68.55694394	74.23915113	95.89117731	79.56242413
26	########	95	0	100	96	76	137	91	--	--	--	--	-1	0	-1	-1--		67.6975588	73.43661354	96.03867	79.05761411
27	########	94	0	99.7	95	76	137	91	--	--	--	--	-1	0	-1	-1--		66.83935097	72.62907715	96.18213995	78.55018936
28	########	93	0	99.8	94	76	137	91	--	--	--	--	-1	0	-1	-1--		65.98302485	71.81596278	96.31972489	78.03957084
29	########	88	0	99.6	94	76	137	91	--	--	--	--	-1	0	-1	-1--		65.12928844	70.99669484	96.44956336	77.52518221
30	########	92	0	99.1	89	76	137	91	--	--	--	--	-1	0	59	-1--		64.27884807	70.17069576	96.5697938	77.00644588

圖二、原始資料(13 場手術)

如圖二所示，原始資料由台大醫院 2014 年期間的 13 場手術所組成，總共 18174 筆資料，且都是使用七氟醚麻醉藥(Sevoflurane)，是目前醫學上其中一種最為常用的麻醉藥。

屬性名稱:HR 為心律(heart rate)，SPO2 為血氧濃度(Saturation of Peripheral Oxygen)，Pulse 為脈搏，NBP 為非侵入式血壓(non-invasive blood pressure)，Sys 為收縮壓(systolic)，Dia 為舒張壓(diastolic)，Mean 為平均值， BIS 為腦電雙頻指數（Bispectral index)，SQI 為訊號品質指標(signal quality index)， EMG 為肌電圖(Electromyography)，還有 A、B、C 三位醫生評估值(Doctor_value)，最後是三位醫生評估的平均值(Average_doctor_value)。PVC、ABP_Sys、ABP_Dia、ABP_Mean、RESP、MAC、SR 沒有任何資料數值，所以不介紹名稱。

不使用隨機劃分或是分層取樣將原始資料(總共 13 場手術)分割出來的測試集，因為其測試集跟訓練集同時擁有同一場手術的資料，導致測試集跟訓練集相似，使其測試結果遠好於原本正確的測試結果。

為了避免此問題發生，所以引入另外 2 場 2014 年的手術當成測試集，總共 1818 筆資料，也都是使用七氟醚麻醉藥(Sevoflurane)。

3.2 處理資料(原始資料跟測試集)

原始資料名稱為 anesthesia(麻醉)，測試集名稱為 anesthesia_test_set_two_surgery。原始資料跟測試集都充滿著-1、”-1”與”- -“的缺值，將這些缺值都取代成 NAN。另外，由於人體生理數值為漸進式，前後的數值很相近，所以可以利用 bfill()函式，將這些 NAN 都取代成該 NAN 值後面最近且非 NAN 的數值，解決缺值的問題。MAC、PVC、ABP_Dia、ABP_Sys、ABP_Mean、RESP、SR 屬性全部都是缺值，所以使用 bfill 方法處理後仍然為缺值，之後會被刪除。

3.3 分層取樣

```
Average_doctor_value    1.000000
C_Doctor_value          0.914894
A_Doctor_value          0.910489
B_Doctor_value          0.851637
BIS                     0.705965
EMG                     0.643608
NBP_Sys                 0.347670
NBP_Mean                0.210769
NBP_Dia                 0.119919
Pulse                   0.095326
HR                      0.090402
SPO2                   -0.118296
ABP_Mean                     NaN
RESP                         NaN
MAC                          NaN
Awareness                    NaN
Name: Average_doctor_value, dtype: float64
```

圖三、原始資料的相關係數

如圖三所示，C_Doctor_value 擁有最高的相

關係數(0.915)，最適合當作分層取樣分群的依據。

將原始資料的 C_Doctor_value 除以 20 以產生 C_Doctor_value_cat 類別，此類別代表原始資料 C_Doctor_value 的分布情況。導入 StratifiedShuffleSplit 函式，將原始資料都打亂混合在一起，並以 C_Doctor_value_cat 類別為依據分割出訓練集與測試集，訓練集與測試集的 C_Doctor_value 分布情況將會與原始資料相同。

如果訓練集和測試集的資料分佈不相同，導致模型在訓練集上的表現很好，而在測試集上卻表現不好；或是導致模型在訓練集上的表現不好，而在測試集上卻表現很好。

原始資料集的 90%為訓練集、10%為測試集，其 10%測試集將會被捨棄，因為使用分層取樣將原始資料(總共 13 場手術)分割出來的訓練集與測試集同時擁有同一場手術的資料，將導致測試集與訓練集相似，使其測試結果遠好於原本正確的測試結果。

為了解決測試集與訓練集含有同一場手術的資料的問題，所以測試集將由另外 2 場 2014 年的手術中提取大部分資料當成新的測試集，且該新測試集與訓練集的資料分布相同。

3.4 檢查新測試集與訓練集和原始資料分佈是否相同

我們希望原始資料、訓練集和測試集的資料分佈相同。如果訓練集和測試集的資料分佈不相同，導致模型在訓練集上的表現很好，而在測試集上卻表現不好；或是導致模型在訓練集上的表現不好，而在測試集上卻表現很好。

新的測試集是由另外 2 場 2014 年的手術中提取大部分資料當成新的測試集。如圖四所示，新測試集的 C_Doctor_value 分佈情況與訓練集和原始資料幾乎相同。

```
# 我們希望測試集、訓練集裡的C_Doctor_value分佈跟原本資料集裡的C_Doctor_value分佈相同，
#此為原本資料集(anesthesia)裡的C_Doctor_value分佈
anesthesia["C_Doctor_value_cat"].value_counts() / len(anesthesia)
```

```
3.0    0.857654
5.0    0.059481
4.0    0.047155
2.0    0.035710
Name: C_Doctor_value_cat, dtype: float64
```

```
# 我們希望測試集、訓練集裡的C_Doctor_value分佈跟原本資料集裡的C_Doctor_value分佈相同，
#此為訓練集(strat_train_set)裡的C_Doctor_value分佈
strat_train_set["C_Doctor_value_cat"].value_counts() / len(strat_train_set)
```

```
3.0    0.857667
5.0    0.059489
4.0    0.047139
2.0    0.035706
Name: C_Doctor_value_cat, dtype: float64
```

```
# 我們希望測試集、訓練集裡的C_Doctor_value分佈跟原本資料集裡的C_Doctor_value分佈相同，
#此為測試集(anesthesia_test_set_two_surgery)裡的C_Doctor_value分佈
anesthesia_test_set_two_surgery["C_Doctor_value_cat"].value_counts() / len(anesthesia_test_set_two_surgery)
```

```
3.0    0.857536
5.0    0.059406
4.0    0.047305
2.0    0.035754
Name: C_Doctor_value_cat, dtype: float64
```

圖四、測試集的分佈情況跟訓練集相同

3.5 設定訓練集、測試集的輸入與標籤

設定訓練集、測試集的輸入與標籤，輸入是 HR、NBP_Sys、EMG、BIS 等等生理數值；標籤為 Average_doctor_value，為三位麻醉醫生評估患者手術中麻醉清醒程度值的平均(0~100)，希望模型模擬麻醉醫生觀看 EMG、BIS 等等生理數值以評估患者手術中麻醉清醒程度。

3.6 自訂轉換器 process_data()

不同的屬性放進模型進行訓練，產生的模型將有不同的預測結果，擁有越多高相關係數屬性，模型預測正確率越高；擁有越多低相關係數屬性，模型預測正確率越低。

分配好訓練集跟測試集後，就需要自訂轉換器來刪除訓練集中相關係數低的屬性，只留下相關係數高的屬性，提升模型的精度。所以設置三組轉換器，將原有訓練集轉換成三組不同訓練屬性的訓練集，並放進模型進行訓練，比較三組模型的預測結果和正確率。

第一組訓練集的訓練屬性: BIS、EMG、NBP_Sys
第二組訓練集的訓練屬性: EMG、NBP_Sys
第三組訓練集的訓練屬性: EMG、NBP_Sys、NBP_Dia、NBP_Mean

如圖五所示，第一組自訂轉換器只留下相關係數前三高的屬性:BIS、EMG、NBP_Sys，刪除訓練集中大部分低相關係數的屬性，形成第一組訓練集。第二組轉換器只留下訓練屬性:EMG、NBP_Sys 為第二組訓練集，第三組轉換器只留下訓練屬性:EMG、NBP_Sys、NBP_Dia、NBP_Mean 為第三組訓練集。

```
from sklearn.base import BaseEstimator,TransformerMixin

class process_data(BaseEstimator, TransformerMixin):
    def fit(self, X, y=None):
        return self  # nothing else to do
    def transform(self, X, y=None):
        X_copy = X.copy()
        #只留下BIS、EMG、NBP_Sys當作訓練參數
        del X_copy ['Time']
        del X_copy ['ABP_Dia']
        del X_copy ['ABP_Sys']
        del X_copy ['ABP_Mean']
        del X_copy ['RESP']
        del X_copy ['SR']
        del X_copy ['NAC']
        del X_copy ['PVC']
        #del X_copy ['BIS']
        #del X_copy ['EMG']
        del X_copy ['SPO2']
        del X_copy ['SQI']
        del X_copy ['Awareness']
        del X_copy ['HR']
        del X_copy ['Pulse']
        del X_copy ['NBP_Dia']
        #del X_copy ['NBP_Sys']
        del X_copy ['NBP_Mean']

        del X_copy ['A_Doctor_value']
        del X_copy ['B_Doctor_value']
        del X_copy ['C_Doctor_value']
        del X_copy ['Average_doctor_value']
        return X_copy
```

```
process_train_set_function = process_data()
anesthesia_process_train_set = process_train_set_function.transform(anesthesia_train_set)
anesthesia_train_parameter = list(anesthesia_process_train_set)
print(anesthesia_train_parameter)
print(anesthesia_process_train_set.head())
```

```
['NBP_Sys', 'BIS', 'EMG']
       NBP_Sys   BIS   EMG
9121      99.0  47.0  27.0
2871     118.0  53.0  31.0
675      116.0  47.0  28.0
4785     137.0  45.0  30.0
11974    114.0  84.0  46.0
```

圖五、第一組自訂轉換器

3.7 制定流水線

如圖六所示，導入 Pipeline 並建立一條流水線，該流水線含有自訂轉換器 process_data()函式與特徵縮放 StandardScaler()函式。原有的訓練集經過此流水線訓練且轉換，原有的訓練集先進行流水線中自訂轉換器 process_data()的資料處理，刪除原始訓練集中低相關性的屬性，只留下重要的屬性，接著執行 Standardization()函式，對訓練集資料進行特徵標準化(特徵縮放)，完成特徵標準化後的訓練集就可以放入各種回歸模型進行訓練。

```
# 建立一個完整的流水線
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
#column

full_pipeline = Pipeline([
        ('process_data', process_data()),
        ('std_scaler', StandardScaler()),
    ])
```

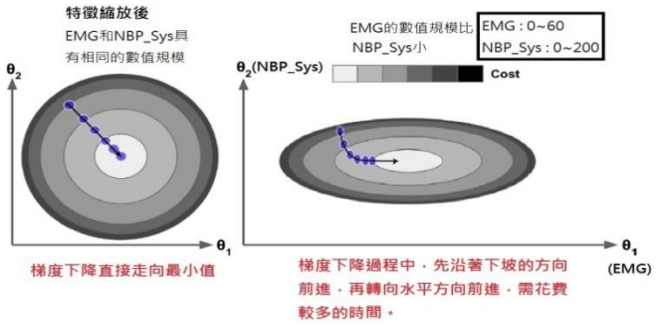
```
anesthesia_train_set_prepared = full_pipeline.fit_transform(anesthesia_train_set)
print(anesthesia_train_set_prepared)
```

```
[[-0.72860525 -0.22246441 -0.48612103]
 [ 0.09032166  0.15975109 -0.05665088]
 [ 0.00411883 -0.22246441 -0.37875349]
 ...
 [ 0.00411883  0.35085884 -0.37875349]
 [ 1.90058113  0.73307433 -0.48612103]
 [-0.25448967 -1.75132639 -0.48612103]]
```

圖六、流水線

特徵縮放又稱特徵標準化，是將特徵資料按比例縮放，讓資料落在某一特定的區間。有兩種特徵標準化方法，第一種為 min max normalization，將特徵數據按比例縮放到 0~1 的區間（或-1~1），第二種為 standard deviation normalization，特徵數據縮放成平均為 0、標準差為 1，本研究使用第二種特徵標準化方法。

如圖七所示，資料中各特徵的數值規模都不一樣，EMG 為 0~60，NBP_Sys 為 0~200，數值規模相差越大，在訓練模型時，梯度下降因為規模不一樣，需花費大量時間才能走向誤差最小值，特徵標準化使特徵擁有相同數值規模，梯度下降直接走向誤差最小值，增進模型收斂速度，減少訓練模型的時間。



圖七、特徵標準化對梯度下降的影響

3.8 回歸模型效能評估方法

藉由評估方法可以分辨模型效能都優劣，因

此能採用最佳的模型，使預測達到最理想化。有兩種評估方法可以評估回歸模型的效能[4]。

第一種評估方法為決定係數（R Squared），代表迴歸模型預測結果與資料答案相符程度的評估計量。R Squared 定義為 SSR 和 SST 的比值，也等於 1 減 SSE 和 SST 比值[4]。

$$R-square=\frac{SSR}{SST}=1-\frac{SSE}{SST}$$

迴歸平方和(Sum of squares of the regression)，即預測值與原始資料均值之差的平方和。

$$SSR=\sum_{i=1}^{n} w_i(\hat{y}_i-\bar{y}_i)^2$$

總平方和(Total sum of squares)，即原始資料和均值之差的平方和[4]。

$$SST=\sum_{i=1}^{n} w_i(y_i-\bar{y}_i)^2$$

殘差平方和(Sum of squares for error)，是原始資料和預測值之差的平方和，又稱和方差[4]。

$$SSE=\sum_{i=1}^{n} w_i(y_i-\hat{y}_i)^2$$

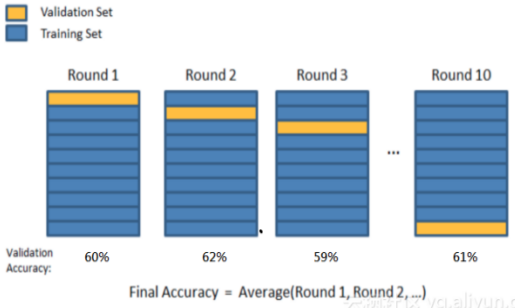
決定係數(R-squared)主要用來測量回歸模型預測結果對訓練集、測試集答案的擬合程度(正確率)，範圍為負值(預測與答案很不相符)到 1(100%正確率->預測與答案完全相同)。在 Python 中，以.score(data，data labels)函式計算模型對訓練集、測試集的決定係數[4]。

第二種評估方法為比較各個模型的訓練誤差、驗證誤差與測試誤差，訓練誤差是預測訓練集的誤差，驗證誤差是交互驗證訓練集的誤差，測試誤差是預測測試集的誤差，通常訓練誤差<驗證誤差<測試誤差。

使用均方根誤差(RMSE)來計算三種誤差，均方根誤差(RMSE)又稱為標準誤差，定義為預測值(ŷ)與實際值(y)偏差的平方與預測值數量比值的平方根，是一種表示預測值與實際值之間誤差的方法[4]。

$$RMSE=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}$$

如圖八所示，交互驗證可以得到驗證誤差，交互驗證是將訓練及分成 10 等份(子集)，其中第 1 等分用來當作驗證的測試資料，其餘 9 份拿來訓練，並且計算均方根誤差，下一輪將第 2 等分拿來當作驗證的測試資料，其餘 9 份一樣拿來訓練，並且計算均方根誤差，以此類推總共做 10 次。總共得到 10 個的均方根誤差(RMSE)，將 10 次平均所得到的平均值，即是驗證誤差。因為輪流測試且平均誤差，所以驗證誤差擁有比訓練誤差更高的可信度[5]。

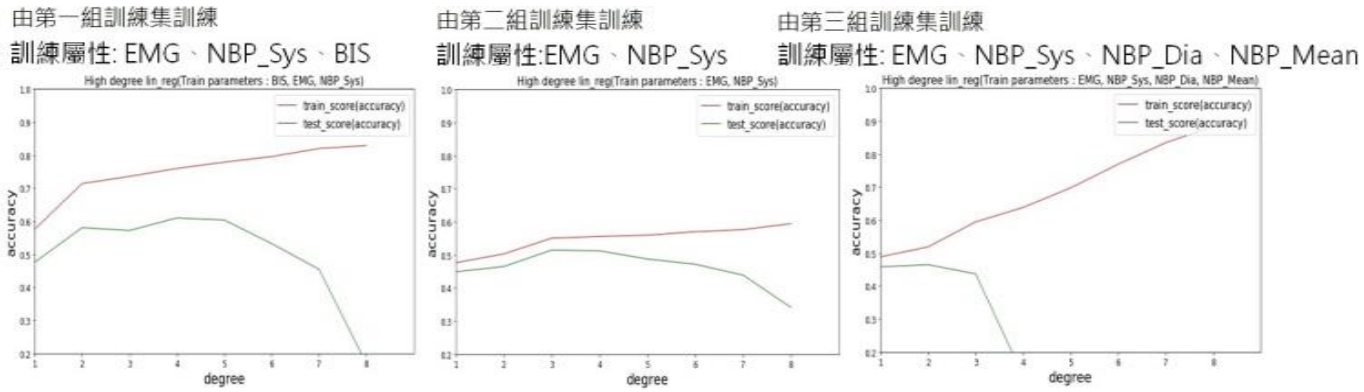


圖八、互驗證(10 個子集)

3.9 線性回歸模型

如圖九所示，讓三組訓練集(不同訓練屬性)訓練一到八階的線性回歸模型，再使用第一種評估方法(決定係數(R-squared))，比較由三組訓練集訓練的一到八階線性回歸模型對訓練集和測試集的決定係數(預測正確率)。

如圖九所示，第一組訓練集訓練的四階線性回歸對測試集的決定係數是最高的(0.61)，代表四階線性模型最適合第一組訓練集；第二組訓練集訓練的三階線性回歸對測試集的決定係數是最高的(0.51)，代表三階線性模型最適合第二組訓練集；第三組訓練集訓練的二階線性回歸對測試集的決定係數是最高的(0.46)，代表二階線性模型最適合第三組訓練集。



圖九、由三組訓練集訓練的一到八階線性回歸模型對訓練集和測試集的決定係數

如圖十所示，使用了二種評估方法，比較各個訓練集的最佳線性模型的決策分數、訓練誤差、驗證誤差和測試誤差。第一組訓練集訓練的最佳的線性模型是四階模型，第二組訓練集的最佳的線性模型是三階模型，第三組訓練集則是二階模型，這三種模型中為四階線性模型擁有最小的測試誤差(5.408)與最高的決定係數(0.61=61%正確率)，代表四階線性模型是最佳的線性回歸模型，也代表第一組訓練集最適合拿來訓練線性回歸模型。

		由第一組訓練集訓練 Four_degree_lin_reg	由第二組訓練集訓練 Three_degree_lin_reg	由第三組訓練集訓練 Two_degree_lin_reg
0	train_score(accuracy)	0.758572	0.549706	0.518547
1	test_score(accuracy)	0.610711	0.514472	0.463970
2	train_set RMSE	5.046915	6.892549	7.127037
3	cross_val_score_mean RMSE	5.076325	6.887337	7.131714
4	test_set RMSE with 3 Doctors average labels	5.408139	7.072344	7.354131

圖十、三組線性回歸模型的效能分析

如圖十一、十二和十三所示，將三組訓練集放入多階線性回歸訓練後，可以利用.intercept_和.coef_得到多階線性回歸模型的常數項跟係數，而.get_feature_names()可以得到係數名稱，將係數名稱、係數與常數項結合，即是多階線性回歸預測資料的公式。如果將測試集中某一時間點的生理數值代入此公式，得到即是模型該時間對患者麻醉清醒程度的預測值。

```
intercept = Four_degree_lin_reg["lin_reg"].intercept_
coef = Four_degree_lin_reg["lin_reg"].coef_
coef_name = Four_degree_lin_reg["poly_features"].get_feature_names(list(anesthesia_process_test_set_two_surgery))

for x in range(34):#coef 的長度為34，所以設定range(34)
    coef[x] = round(coef[x],2)#取到小數點後兩位

print("Four_degree_lin_reg")
print("y =",intercept,"+",coef[0],coef_name[0],"+",coef[1],coef_name[1],"+",coef[2],coef_name[2],"+",coef[3],coef_name[3],
      "+",coef[4],coef_name[4],"+",coef[5],coef_name[5],"+",coef[6],coef_name[6],"+",coef[7],coef_name[7],
      "+",coef[8],coef_name[8],"+",coef[9],coef_name[9],"+",coef[10],coef_name[10],"+",coef[11],coef_name[11],
      "+",coef[12],coef_name[12],"+",coef[13],coef_name[13],"+",coef[14],coef_name[14],"+",coef[15],coef_name[15],
      "+",coef[16],coef_name[16],"+",coef[17],coef_name[17],"+",coef[18],coef_name[18],"+",coef[19],coef_name[19],
      "+",coef[20],coef_name[20],"+",coef[21],coef_name[21],"+",coef[22],coef_name[22],"+",coef[23],coef_name[23],
      "+",coef[24],coef_name[24],"+",coef[25],coef_name[25],"+",coef[26],coef_name[26],"+",coef[27],coef_name[27],
      "+",coef[28],coef_name[28],"+",coef[29],coef_name[29],"+",coef[30],coef_name[30],"+",coef[31],coef_name[31],
      "+",coef[32],coef_name[32],"+",coef[33],coef_name[33])
```

```
Four_degree_lin_reg
y = 44.210589896617655 + 1.24 NBP_Sys + -0.12 BIS + 0.64 EMG + 1.78 NBP_Sys^2 + 0.59 NBP_Sys BIS + 0.09 NBP_Sys EMG + 3.72 BIS^
2 + -0.85 BIS EMG + -0.02 EMG^2 + -0.11 NBP_Sys^3 + 1.21 NBP_Sys^2 BIS + -0.2 NBP_Sys^2 EMG + 0.58 NBP_Sys BIS^2 + 0.44 NBP_Sys
BIS EMG + 0.08 NBP_Sys EMG^2 + -0.31 BIS^3 + 1.47 BIS^2 EMG + 1.08 BIS EMG^2 + -0.19 EMG^3 + -0.09 NBP_Sys^4 + -0.14 NBP_Sys^3
BIS + 0.14 NBP_Sys^3 EMG + -0.87 NBP_Sys^2 BIS^2 + 0.64 NBP_Sys^2 BIS EMG + -0.18 NBP_Sys^2 EMG^2 + 0.06 NBP_Sys BIS^3 + -0.29
NBP_Sys BIS^2 EMG + -0.01 NBP_Sys BIS EMG^2 + -0.12 NBP_Sys EMG^3 + -0.19 BIS^4 + 0.33 BIS^3 EMG + -0.57 BIS^2 EMG^2 + -0.3 BIS
EMG^3 + 0.13 EMG^4
```

圖十一、第一組訓練集的四階線性回歸預測公式

```
intercept = Three_degree_lin_reg["lin_reg"].intercept_
coef = Three_degree_lin_reg["lin_reg"].coef_
coef_name = Three_degree_lin_reg["poly_features"].get_feature_names(list(anesthesia_process_test_set_two_surgery))

for x in range(9):#coef 的長度為9，所以設定range(9)
    coef[x] = round(coef[x],2)#取到小數點後兩位

print("Three_degree_lin_reg")
print("y =",intercept,"+",coef[0],coef_name[0],"+",coef[1],coef_name[1],"+",coef[2],coef_name[2],"+",coef[3],coef_name[3],
      "+",coef[4],coef_name[4],"+",coef[5],coef_name[5],"+",coef[6],coef_name[6],"+",coef[7],coef_name[7],
      "+",coef[8],coef_name[8])
```

```
Three_degree_lin_reg
y = 46.65897397663362 + 4.08 NBP_Sys + 4.93 EMG + 1.29 NBP_Sys^2 + 3.48 NBP_Sys EMG + 1.98 EMG^2 + -0.51 NBP_Sys^3 + -0.04 NBP_
Sys^2 EMG + -0.84 NBP_Sys EMG^2 + -0.56 EMG^3
```

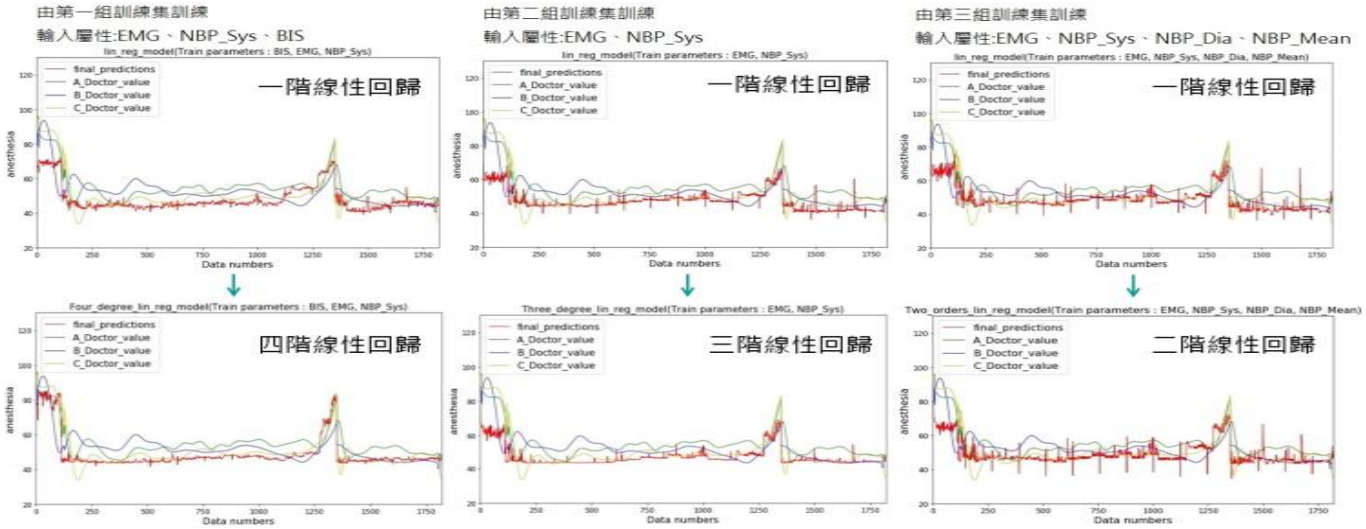
圖十二、第二組訓練集的三階線性回歸預測公式

```
intercept = Two_degree_lin_reg["lin_reg"].intercept_
coef = Two_degree_lin_reg["lin_reg"].coef_
coef_name = Two_degree_lin_reg["poly_features"].get_feature_names(list(anesthesia_process_test_set_two_surgery))

for x in range(14):#coef 的長度為14，所以設定range(14)
    coef[x] = round(coef[x],2)#取到小數點後兩位

print("Two_degree_lin_reg")
print("y =",intercept,"+",coef[0],coef_name[0],"+",coef[1],coef_name[1],"+",coef[2],coef_name[2],"+",coef[3],coef_name[3],
      "+",coef[4],coef_name[4],"+",coef[5],coef_name[5],"+",coef[6],coef_name[6],"+",coef[7],coef_name[7],
      "+",coef[8],coef_name[8],"+",coef[9],coef_name[9],"+",coef[10],coef_name[10],"+",coef[11],coef_name[11],
      "+",coef[12],coef_name[12],"+",coef[13],coef_name[13])
```

```
Two_degree_lin_reg
y = 48.50161951225392 + -5.17 NBP_Dia + 0.83 NBP_Sys + 5.44 NBP_Mean + 7.15 EMG + 3.4 NBP_Dia^2 + -2.3 NBP_Dia NBP_Sys + -3.17
NBP_Dia NBP_Mean + -1.41 NBP_Dia EMG + -1.65 NBP_Sys^2 + 6.4 NBP_Sys NBP_Mean + 0.0 NBP_Sys EMG + -1.57 NBP_Mean^2 + 2.31 NBP_M
ean EMG + -0.63 EMG^2
```

圖十三、第三組訓練集的二階線性回歸預測公式

如圖十四所示，讓各種線性回歸模型去預測測試集，且將預測值視覺化與三位醫生評估值比較。四階線性回歸測試正確率最高(61%)，其模型的紅色預測曲線也跟三位醫生評估的曲線相似度最高。



圖十四、多種線性回歸模型的預測曲線

3.10 調整超參數後的決策樹回歸模型

對三組訓練集訓練的決策樹模型進行網格收尋（GridSearchCV），在所有候選的超參數選擇中，通過循環遍歷，嘗試每一種參數選擇，找尋最佳的超參數，找尋最佳的決策樹模型，此過程稱為『調參』。

如圖十五所示，"criterion"為特徵選擇標準，回歸模型推薦使用'mse',"splitter"為特徵劃分點選擇標準，設定為'best'，在特徵的所有劃分點中找出最優的劃分點，"max_depth"為決策樹的最大深度，設定最大深度可以防止過度擬合，"max_leaf_nodes"為最大葉子節點數目，限制最大葉子節點數可以防止過度擬合。

將上述的超參數設定一個超參數列表，讓三組由不同訓練集訓練的決策樹模型嘗試列表中的選擇，找出各組模型最好的超參數，即是找出各組最好的決策樹模型。

如圖十六所示，使用了二種評估方法，比較三組調參後的決策樹模型的決策分數、訓練誤差、驗證誤差和測試誤差。三種模型中由第一組訓練集訓練的調參決策樹模型擁有最小的測試誤差(5.8)與最高的決定係數(0.55=55%正確率)，為最佳調參決策樹模型，也代表第一組訓練集是最適合放進決策樹回歸模型訓練的訓練集。

```
from sklearn.model_selection import GridSearchCV

param_grid = [{"criterion":['mse'],
               "splitter":['best'],
               "max_depth": [2,3,4,5,6,7,8],
               "max_leaf_nodes":[2,3,4,5,6,7,8],}]

tree_reg = DecisionTreeRegressor(random_state=42)
tree_reg_grid_search = GridSearchCV(tree_reg, param_grid, cv=5, return_train_score=True,n_jobs=4)
tree_reg_grid_search.fit(anesthesia_train_set_prepared, anesthesia_train_set_labels)
print('')
```

```
# 顯示最佳的超參數組合
Optimal_tree_reg_model = tree_reg_grid_search.best_params_
print(Optimal_tree_reg_model)
```

```
{'max_depth': 3, 'max_leaf_nodes': 8}
```

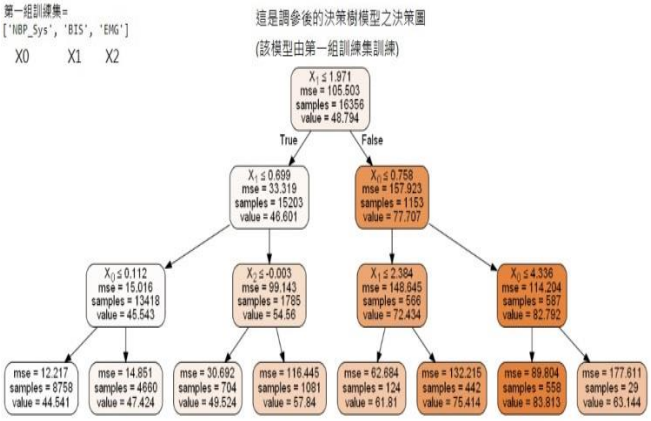
圖十五、尋找由第一組訓練集訓練的模型的最佳超參數

		由第一組訓練集訓練且調參 Optimal_tree_reg	由第二組訓練集訓練且調參 Optimal_tree_reg	由第三組訓練集訓練且調參 Optimal_tree_reg
0	train_score(accuracy)	0.742027	0.602296	0.606813
1	test_score(accuracy)	0.550286	0.409250	0.398448
2	train_set RMSE	5.216976	6.477563	6.440673
3	cross_val_score_mean RMSE	5.228822	6.528629	6.545925
4	test_set RMSE with 3 Doctors average labels	5.802235	6.650116	6.710637

圖十六、三組調參後的決策樹模型的效能分析

如圖十七所示，將最佳的調參決策樹模型視覺化，可以了解決策樹如何預測數值，使用 export_graphviz 導出器導出的決策樹規則(dot 物件)，經過 pydotplus 將決策樹規則解析為決策圖，最後用 Image 將決策圖顯示在 Jupyter

Notebook 上。設定 filled=True 讓 value 越大者節點框。顏色越深；設定 rounded=True 以繪製帶圓角的[6]。



圖十七、最佳的調參決策樹模型之決策圖，該模型由第一組訓練集訓練，該訓練集有三個屬性:(a)X0 為 NBP_Sys; (b) X1 為 BIS; (c)X2 為 EMG

如圖十八所示，讓三組調參後的決策樹模型去預測測試集，且將預測值視覺化與三位醫生評估值比較。第一組訓練集訓練的模型的測試正確率最高(0.55=55%)，其模型的紅色預測曲線也跟三位醫生評估的曲線相似度最高。

3.11 超參數優化的隨機森林回歸模型

如圖十九所示，對隨機森林模型進行網格收尋（GridSearchCV），在所有候選的超參數選擇中，通過循環遍歷，嘗試每一種參數選擇，找尋最佳的超參數，找尋最佳的隨機森林模型。介紹隨機森林的超參數，"bootstrap":True，為隨機採樣以訓練森林中每一棵決策樹，訓練完放回樣本；"max_features：auto、sqrt、log2"為每棵決策樹最大的特徵數量；"n_estimators"為森林裡決策樹的數量，通常越多越好；"max_depth"為每棵決策樹的最大深度，設定最大深度可以防止過度擬合；"max_leaf_nodes"為每棵決策樹最大葉子節點數目，限制最大葉子節點數可以防止過度擬合。

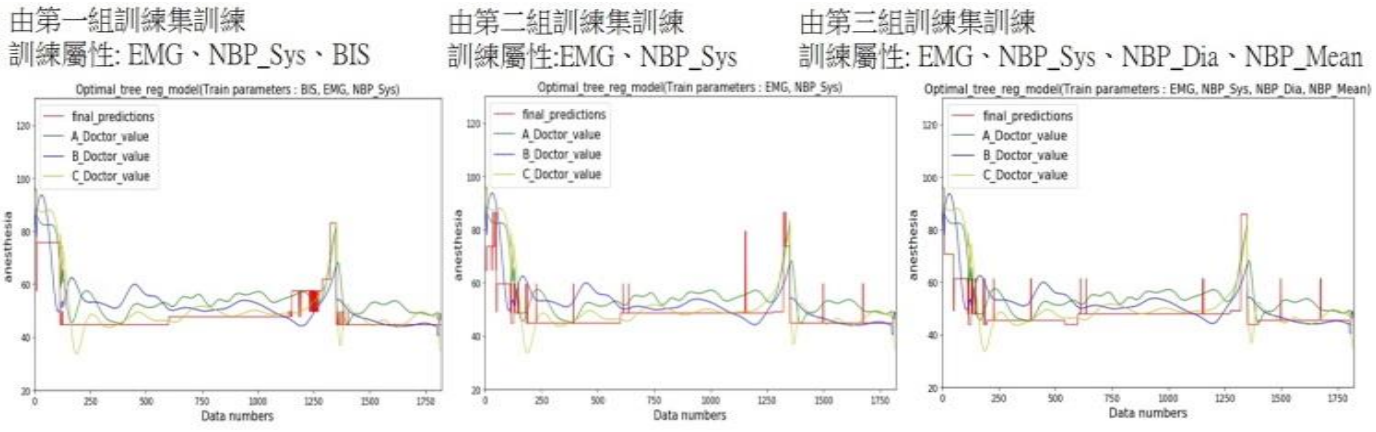
將上述的超參數設定一個超參數列表，讓三組由不同訓練集訓練的隨機森林模型嘗試列表中的選擇，找出各組模型最好的超參數，即是找出各組最好的隨機森林模型。

```
from sklearn.model_selection import GridSearchCV

param_grid = [
    # 嘗試 1620種 (2×3×3×3×6×5) 超參數組合
    {'bootstrap':[True,False],
     'max_features': ['auto','sqrt','log2'],
     'n_estimators': [200,300,400],
     'max_depth':[7,8,9,10,11,12],
     'max_leaf_nodes':[4,5,6,8,9],
    },]
forest_reg_Optimal_model = RandomForestRegressor(random_state=42)
#交叉驗證5次, 總共 (1620)*5=8100次 的訓練
Optimal_forest_reg_grid_search = GridSearchCV(forest_reg_Optimal_model, param_grid, cv=5,
                          scoring='neg_mean_squared_error', return_train_score=True,n_jobs=2)
Optimal_forest_reg_grid_search.fit(anesthesia_train_set_prepared, anesthesia_train_set_labels)
```

```
# 顯示最佳的超參數組合
Optimal_forest_reg_grid_search.best_params_
```

```
{'bootstrap': True,
 'max_depth': 8,
 'max_features': 'sqrt',
 'max_leaf_nodes': 8,
 'n_estimators': 300}
```

圖十九、尋找由第一組訓練集訓練的模型的最佳超參數



圖十八、三組調參決策樹模型的預測曲線

如圖二十所示，使用了二種評估方法，比較各組訓練集訓練的調參隨機森林模型的決策分數、訓練誤差、驗證誤差和測試誤差。

三組模型中，由第一組訓練集訓練的模型擁有最小的測試誤差(5.418)與最高的決定係數(0.607)，代表由第一組訓練集訓練的隨機森林模型是最佳的模型，也代表第一組訓練集是最適合放進隨機森林模型訓練的訓練集。

	由第一組訓練集訓練且調參 Optimal_forest_reg	由第二組訓練集訓練且調參 Optimal_forest_reg	由第三組訓練集訓練且調參 Optimal_forest_reg
train_score(accuracy)	0.765590	0.672212	0.689879
test_score(accuracy)	0.607751	0.520813	0.486103
train_set RMSE	4.973020	5.880690	5.720022
cross_val_score_mean RMSE	4.995510	5.926112	5.767689
test_set RMSE with 3 Doctors average labels	5.418866	5.989352	6.202480

圖二十、三組調參隨機森林模型的效能分析

如圖二十一所示，讓三組調參後的隨機森林模型去預測測試集，且將預測值視覺化與三位醫生評估值比較。第一組訓練集訓練的模型的測試正確率最高(0.607=61%)，其模型的紅色預測曲線也跟三位醫生評估的曲線最相似。

3.12 最佳的回歸模型

如圖二十二所示，從全部模型中列出前四名的回歸模型，並使用二種評估方法比較四種模型的決策分數、訓練誤差、驗證誤差和測試誤差。四種模型中有三種都是由第一組訓練集訓練，代表第一組訓練集是三組訓練集中最佳的，擁有最高相關的屬性，訓練出來的模型最準確。

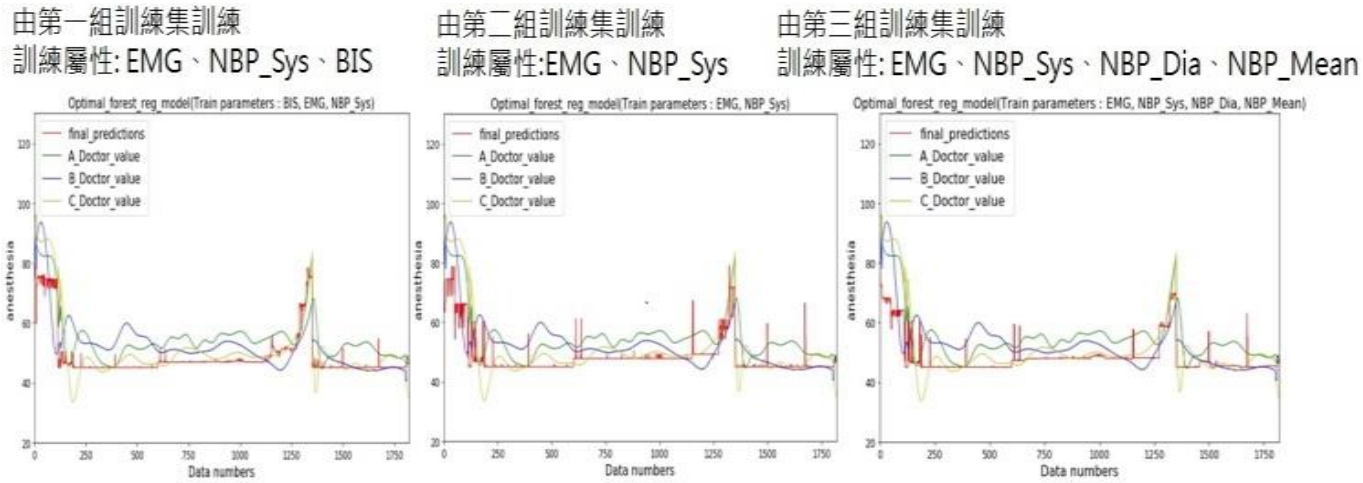
第一名模型為四階線性回歸模型，該模型的對測試集的均方根誤差最低(5.4)，該模型對測試集的決定係數也最高(0.61=61%)，代表該模型對測試集預測的正確度最高(61%)，此模型將做為在臨床上執行預測的模型。

	由第一組訓練集訓練 Four_degree_lin_reg	由第一組訓練集訓練 Optimal_tree_reg	由第一種訓練集訓練 Optimal_forest_reg	由第二組訓練集訓練 Optimal_forest_reg
train_score(accuracy)	0.758572	0.742027	0.765590	0.672212
test_score(accuracy)	0.610711	0.550286	0.607751	0.520813
train_set RMSE	5.046915	5.216976	4.973020	5.880690
cross_val_score_mean RMSE	5.076325	5.228822	4.995510	5.926112
test_set RMSE with 3 Doctors average labels	5.408139	5.802235	5.418866	5.989352
	第一名	第三名	第二名	第四名

圖二十二、前四名的回歸模型

3.13 資料處理流水線與最佳模型結合

將 3.7 小節的資料處理流水線與最佳模型(四階線性回歸)結合成一個最終的流水線，最終流水線具備資料處理與模型訓練與預測功能，只要將訓練集經過最終流水線訓練完畢(完成資料處理與訓練四階線性回歸模型)，最終流水線即可預測測試資料中某時間點病人清醒程度狀態。



圖二十一、三組調參隨機森林模型的預測曲線

3.14 保存模型並直接預測

利用 joblib.dump()將訓練好的最終流水線完整保存於 pkl 檔中。

如圖二十三所示，使用 joblib.load()載入 pkl 檔中的預測器(四階線性回歸模型)，使用.predict()讓預測器直接預測測試集。

```
predictions = my_model_loaded.predict(anesthesia_test_set_two_surgery)
print("predictions = ",predictions)
```

```
predictions =  [75.84134529 78.13348082 77.0054833  ... 46.58829216 45.41697757
 45.69402736]
```

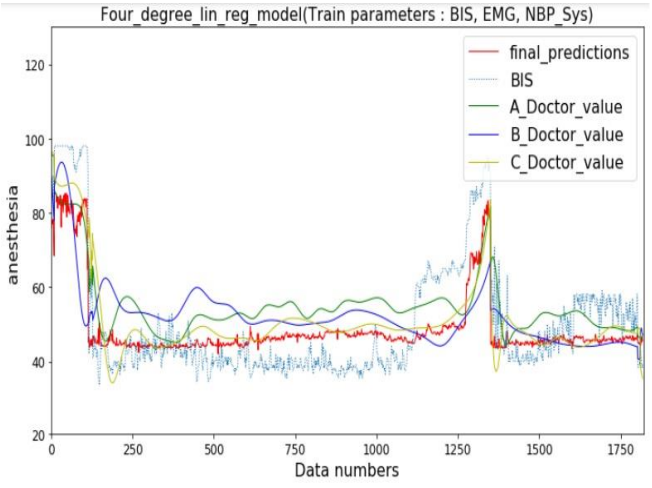
```
import numpy
labels = np.array([])
for x in anesthesia_test_set_two_surgery_labels:
    labels = np.append(labels,x)
print("labels =",labels)
```

```
labels = [93.80663633 93.27216914 92.74693475 ... 43.26921621 43.16681072
 43.08013558]
```

圖二十三、模型之預測值與標籤(答案)的比較

3.15 最佳模型與原有評估方法(BIS)比較

如圖二十四所示，四階線性回歸模型對測試集進行預測且視覺化，模型預測曲線比現有臨床上其中一種評估方法(BIS)更接近三位醫生的評估曲線，且模型預測曲線也與三位醫生的評估曲線非常相近，代表該模型的預測評估結果與麻醉醫師的真人評估結果很相似，成功提供執刀醫生更準確的病人清醒程度，也成功優化現有臨床上的腦波雙頻指數(BIS)評估方法。



圖二十四、BIS 與最佳模型在測試集的比較

四、結果與討論

本論文嘗試以三組訓練集訓練各種回歸演算法，且讓各種模型演算法模擬三位醫生評估病人手術中的清醒程度，使用兩種回歸模型效能評估方法來評估各種模型，為決定係數與均方根誤差(RMSE)，透過分析各模型的決定係數與均方根誤差(RMSE)，從全部模型中找出前四名的模型，發現其中三名都由第一組訓練集訓練，代表第一組訓練集是三組訓練集中最佳的，擁有最高相關的屬性。

將第一名模型當作最終臨床預測的模型，且對測試集進行預測並且視覺化，發現模型預測曲線比現有臨床上其中一種評估方法(BIS)更接近三位醫生的評估曲線，且模型預測曲線也與三位醫生的評估曲線非常相近，代表該模型的預測評估結果與麻醉醫師的真人評估結果很相似，成功提供執刀醫生更準確的病人清醒程度。

參考文獻

[1]American Society of Anesthesiologists and the American Association of Nurse Anesthetists, “Patient awareness under general anesthesia—what is it?”, 2005

[2]秋怡芳，鎮痛傷害性指數於全身麻醉病人插管前後疼痛評估之研究，Analgesia/Nociception Index Applied to Pain Assessment of Intubation During General Anaesthesia，碩士論文，元智大學，2015

[3]李定達，應用機器學習演算法識別罹患帶狀泡疹之高危險群，Applying Machine Learning Algorithms to Identify Patients with High Risk of Developing Herpes Zoster，博士論文，台灣大學，2013

[4]Sebastian Raschka; Vahid Mirjalili; 劉立民、吳建華譯, “Python 機器學習 第二版”, 博碩 , 2018: p. 318-321

[5] https://ithelp.ithome.com.tw/articles/10197461

[6]Aurélien Géron, “Hands-On Machine Learning with Scikit-Learn & TensorFlow”, O'Reilly , 2017: p. 125、168-173