HW4-Team11 Arteon

陳泓吟 工資 112 H34086254
陳宜姍 統計 114 C14101137
陳凱騫 統計 114 H24101222

1.競賽敘述與目標

此次的競賽目標為銀行客戶的流失預測。在原始資料中有 14 個欄位，其中包含一個欄位 Exited 為目標欄位代表客戶是否流失，以 0、1 表示(0 代表客戶無流失，1 代表客戶流失)。競賽目標為針對剩餘 13 個欄位判斷其對於客戶是否流失的關聯性，進行特徵的刪減或增加，並嘗試各種不同的預測模型預測該客戶是否流失，達到最好的預測結果。

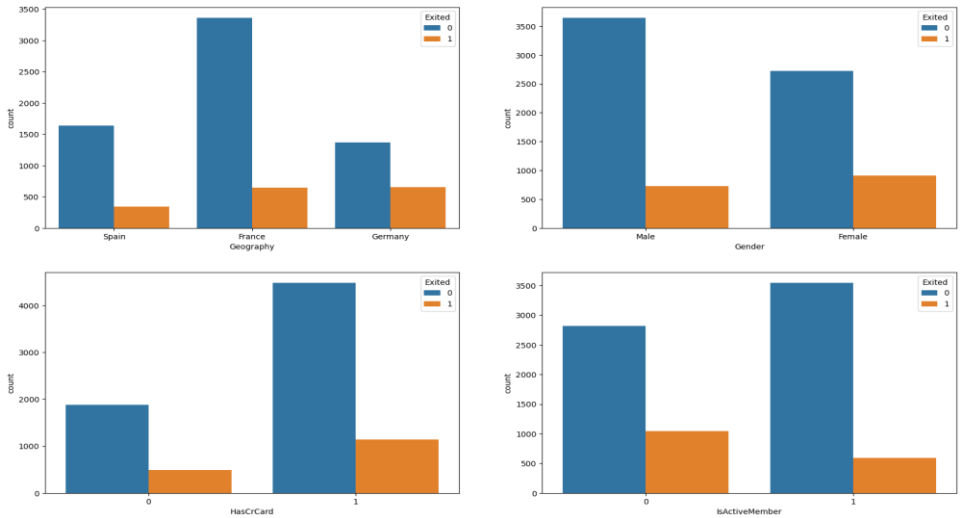
2.資料前處理

根據常理判斷，我們認為 RowNumber、CustomerId、Surname 三個欄位對客戶是否會流失較無影響，因此刪除過濾掉這三個欄位。

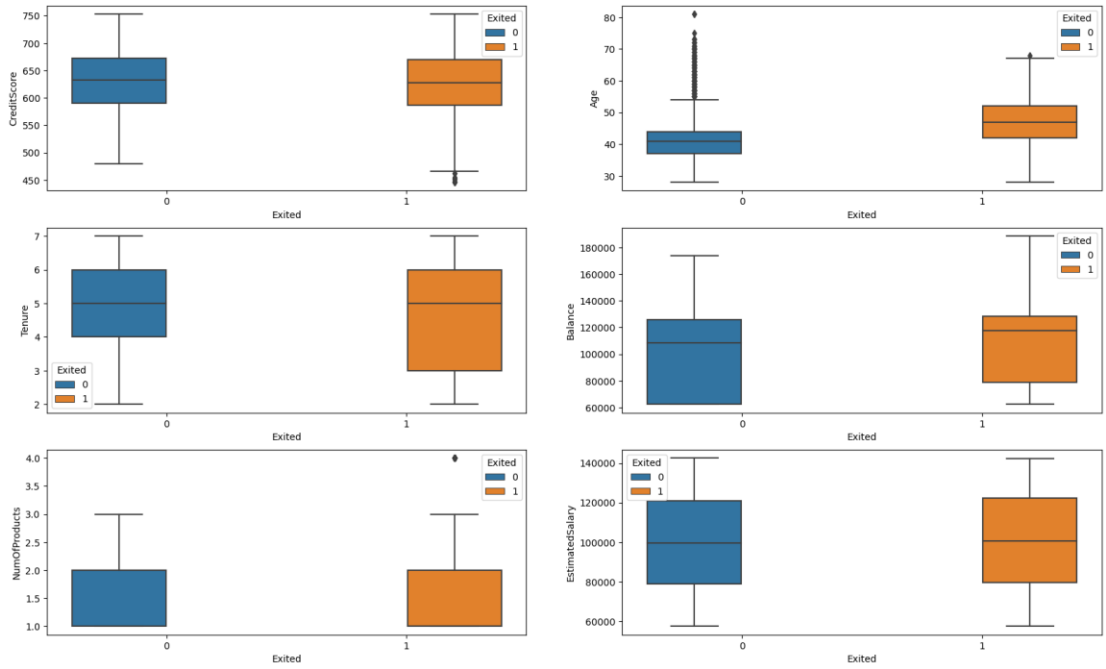
3.特徵處理與分析

3.1 觀察資料分布情形

首先我們有參考 kaggle，此處我們下載了完整的 Churn_Modelling.csv 10000 筆資料，針對原始資料中的不同欄位進行視覺化，確認其資料分布。如下圖(一)、(二)。由圖(一)、(二)中可看出 CreditScore、NumOfProducts、EstimatedSalary 三個連續型資料欄位在客戶是否流失的數據分布上沒有明顯的差異。在之後的預測模型訓練中可考慮針對這三個欄位進行刪減或與其他欄位合併計算產生新特徵。



(圖一) 類別型變數欄位資料分布



圖(二) 連續型變數欄位資料分布情形

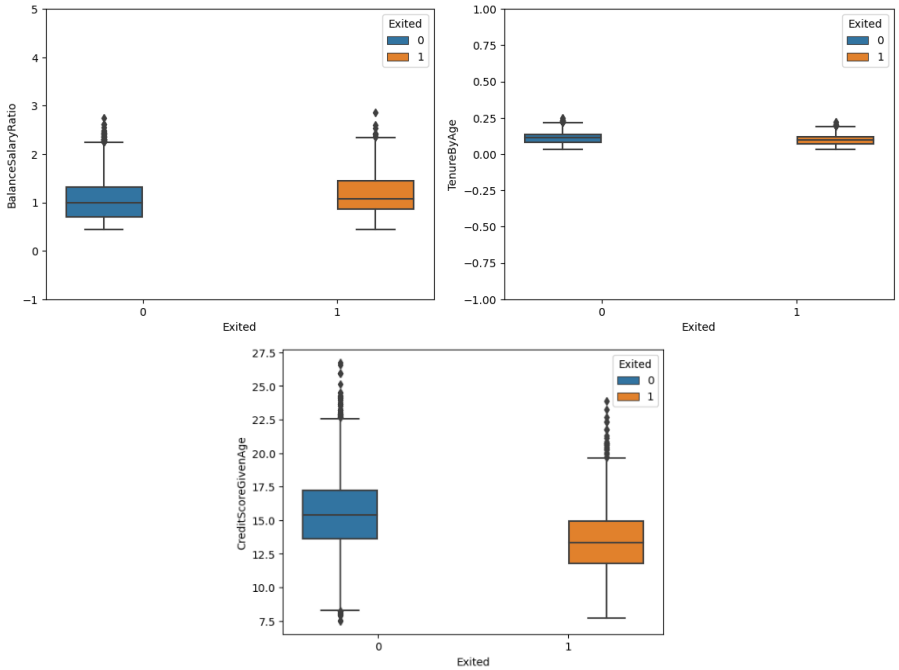
3.2 增加欄位

根據 kaggle 建議，我們在現有的欄位中再自行增加三個欄位，並且同樣匯出資料分布圖如圖(四)。此三個欄位分別為:

BalanceSalaryRatio=Balance/EstimatedSalary

TenureByAge=Tenure/Age

CreditScoreGivenAge=CreditScore/Age



圖(四) 增加欄位的數據分布情祥

根據圖(四)可看出增加的三個欄位在客戶是否流失的數據分布上有明顯的差異，尤其是 EstimatedSalary、CreditScore 兩欄位在與其他欄位進行合併計算產生出的新欄位 BalanceSalaryRatio 及 CreditScoreGivenAge 改善了原先其在客戶是否流失的數據分布上沒有明顯的情形。

3.3 非連續型資料欄位處理

原始資料中屬於類別型資料的有 Geography、Gender 以及分別以 0、1 表示的 HasCard、IsActiveMember。針對 Geography、Gender 欄位使用 get_dummies 將欄位轉為圖(五)

Geography_France	Geography_Germany	Geography_Spain	Gender_Female	Gender_Male
0	0	1	0	1
1	0	0	1	0
1	0	0	1	0
0	0	1	1	0
0	0	1	0	1

圖(五) Geography、Gender 欄位轉換

將 Geography、Gender 轉成如圖(五)所示後，再加上 HasCard、IsActiveMember 共 7 個欄位原先有再轉換成以 1、-1 表示，如圖(六)，以-1 代替原先的 0 或許可以提升訓練結果。但後來實際以同一套模型測試轉換成 1、-1 的預測準確率並無提升，因此之後的模型就沒有再進行該轉換。

HasCrCard	IsActiveMember	Geography_Spain	Geography_Germany	Geography_France	Gender_Male	Gender_Female
1	-1	1	-1	-1	1	-1
1	1	-1	1	-1	1	-1
-1	1	-1	-1	1	-1	1
1	1	-1	-1	1	-1	1
1	1	-1	-1	1	1	-1

圖(六) 非連續型資料欄位轉換

3.4 連續型資料處理

針對連續型資料欄位，我們使用 StandardScaler 將所有連續型特徵進行標準化。轉換成如圖(七)所示。

	CreditScore	Age	Tenure	Balance	NumOfProducts	EstimatedSalary	BalanceSalaryRatio	TenureByAge	CreditScoreGivenAge
0	1.609183	-0.508456	-1.222346	-1.272730	-0.889646	0.263584	-1.116740	-0.902418	1.220540
1	0.528128	-0.112514	-1.222346	0.915085	-0.889646	1.077935	-0.120174	-1.058773	0.218768
2	-1.401138	-1.168361	1.184637	-1.272730	0.841183	-0.887917	-0.653827	1.930377	0.320194
3	1.692341	-0.112514	-1.222346	-0.489610	-0.889646	1.502999	-1.006459	-1.058773	0.799043
4	-1.600717	0.943333	1.184637	-1.272730	-0.889646	1.468965	-1.397624	0.323410	-1.549248

圖(七) 連續型資料欄位轉換

4.預測訓練模型

在預測模型部分，我們嘗試了 DecisionTreeClassifier、RandomForestClassifier、XGBClassifier 、 ExtraTreesClassifier 、 LogisticRegression 、 CatBoostClassifier 、 BaggingClassifier、Boosting、VotingClassfier 等等不同的模型。

DecisionTreeClassifier

DecisionTreeClassifier 的部分沒有特別使用 GridSearchCV，而是直接選定不同的參數範圍進行人工調整，max_depth 的測試範圍有 5、8、10，criterion=gini、entropy。並且再套上 BaggingClassifier，BaggingClassifier 的參數測試範圍如下表。

n_estimators	30、50、100
max_features	0.3、0.5、0.7、0.9

最終選擇的參數如下:

n_estimators	100
max_features	0.7

而我們在該參數下得到我們最佳的預測結果。

```
clf = tree.DecisionTreeClassifier(criterion= 'entropy',max_depth= 8)
```

```
bag = BaggingClassifier(base_estimator= clf,n_estimators= 100,max_features= 0.7,n_jobs= -1)
```

另外在此我們也有利用 RFE 進行特徵選擇，結果顯示 n_features_to_select=15，但將此結果 fit 在訓練資料再進行模型預測後的結果並沒有比較好。

RandomForestClassifier

以 GridSearchCV 測試 RandomForestClassifier，測試的參數範圍如下表:

max_depth	[3, 5, 6, 7, 8]
max_features	[2,4,6,7,8,9]
n_estimators	[50,100]
min_samples_split	[3, 5, 6, 7]

最終選擇的參數如下:

max_depth	8
max_features	6
n_estimators	50
min_samples_split	3

針對上表的參數，RandomForestClassifier 也有嘗試各種不同變化

- 單純使用 RandomForestClassifier
- 先使用 PCA 降維，n_components = 8、9、10，再套用 RandomForestClassifier

- RandomForestClassifier + BaggingClassifier(base_estimator= RF,n_estimators= 100,max_features= 0.7,n_jobs= -1) (此處 BaggingClassifier 的參數是參考其與 DecisionTreeClassifier 並用時最好的參數)

GradientBoosting

以 GridSearchCV 測試 GradientBoosting，測試的參數範圍如下表:

n_estimators	[50,100]
learning_rate	[0.3,0.5,0.7]
max_depth	[5,7,9]

最終選擇的參數如下:

n_estimators	50
learning_rate	0.3
max_depth	5

XGBClassifier

以 GridSearchCV 測試 XGBClassifier，測試的參數範圍如下表:

max_depth	[5,6,7,8]
gamma	[0.01,0.001,0.001]
learning_rate	[0.05,0.1, 0.2, 0.3]

最終選擇的參數如下:

max_depth	7
gamma	0.01
learning_rate	0.1

ExtraTreesClassifier

以 GridSearchCV 測試 ExtraTreesClassifier，測試的參數範圍如下表:

n_estimators	range(10,100,10)
max_depth	range(1,10)

最終選擇的參數如下:

n_estimators	90
max_depth	9

針對上表的參數，ExtraTreesClassifier 也有嘗試各種不同變化

- 單純使用 ExtraTreesClassifier
- 除了 GridSearchCV 的參數外再選擇另一套不同的參數，並利用 VotingClassifier 併用。

LogisticRegression

以 GridSearchCV 測試 LogisticRegression，測試的參數範圍如下表:

C	[1,10,50,100]
tol	[0.00001,0.0001,0.000001]

最終選擇的參數如下:

C	100
tol	0.00001

CatBoostClassifier

以 GridSearchCV 測試 CatBoostClassifier，測試的參數範圍如下表:

iterations	[100, 150, 200]
learning_rate	[0.03, 0.1]
depth	[2, 4, 6, 8]
l2_leaf_reg	[0.2, 0.5, 1, 3]

最終選擇的參數如下:

iterations	150
learning_rate	0.1
depth	4
l2_leaf_reg	0.5

以上所有的模型除了嘗試 GridSearchCV 所決定出的參數外，也有自行跑其他不同的參數組合，另外也有再結合 BaggingClassifier、VotingClassfier、StackingClassifier 進行不同的模型訓練。

5.預測結果分析

上述在進行模型及各個參數選擇時主要是使用 train.csv 的原始資料進行切割測試，切割成 0.8 的訓練資料及 0.2 的測試資料。透過測試資料檢視各模型的預測準確率。因為上傳的預測結果眾多，因此以下的預測結果分析會先放上最佳預測模型自行測試的結果及上傳網站後的結果，而其他的模型則挑選特定幾筆紀錄放上上傳網站後的結果。

最佳預測模型

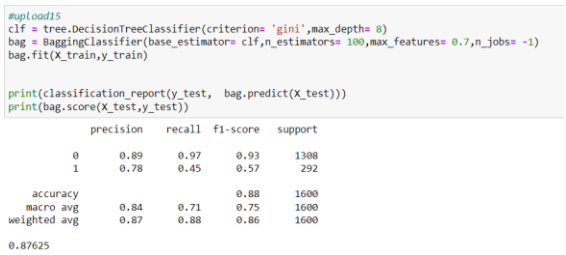
下圖(八)是單用 DecisionTreeClassifier 自行測試的結果

```
clf = tree.DecisionTreeClassifier(criterion= 'entropy',max_depth= 8)

clf.fit(X_train,y_train)

from sklearn.metrics import classification_report
print(classification_report(y_test,  clf.predict(X_test)))
print(clf.score(X_test,y_test))
```

```
              precision    recall  f1-score   support

           0       0.87      0.94      0.91      1277
           1       0.68      0.46      0.55       323

    accuracy                           0.85      1600
   macro avg       0.78      0.70      0.73      1600
weighted avg       0.83      0.85      0.84      1600

0.846875
```

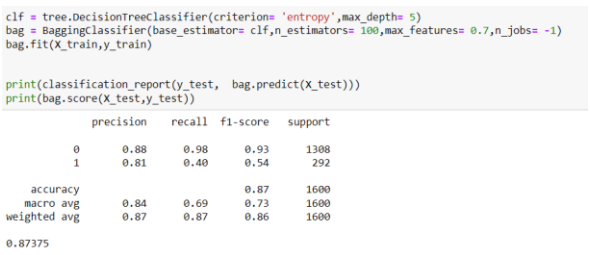
圖(八) DecisionTreeClassifier 模型測試

下圖(九)是其中兩筆 DecisionTreeClassifier+BaggingClassifier 調用不同參數自行測試的結果



```
#upload15
clf = tree.DecisionTreeClassifier(criterion= 'gini',max_depth= 8)
bag = BaggingClassifier(base_estimator= clf,n_estimators= 100,max_features= 0.7,n_jobs= -1)
bag.fit(X_train,y_train)


print(classification_report(y_test,  bag.predict(X_test)))
print(bag.score(X_test,y_test))
```

```
              precision    recall  f1-score   support

           0       0.89      0.97      0.93      1308
           1       0.78      0.45      0.57       292

    accuracy                           0.88      1600
   macro avg       0.84      0.71      0.75      1600
weighted avg       0.87      0.88      0.86      1600

0.87625
```



```
clf = tree.DecisionTreeClassifier(criterion= 'entropy',max_depth= 5)
bag = BaggingClassifier(base_estimator= clf,n_estimators= 100,max_features= 0.7,n_jobs= -1)
bag.fit(X_train,y_train)


print(classification_report(y_test,  bag.predict(X_test)))
print(bag.score(X_test,y_test))
```

```
              precision    recall  f1-score   support

           0       0.88      0.98      0.93      1308
           1       0.81      0.40      0.54       292

    accuracy                           0.87      1600
   macro avg       0.84      0.69      0.73      1600
weighted avg       0.87      0.87      0.86      1600

0.87375
```

圖(九) DecisionTreeClassifier+BaggingClassifier 模型測試

而圖(十)為我們最佳預測結果的模型參數，圖(十一)為測試結果。表格則為上傳網站後的結果。

```
clf = tree.DecisionTreeClassifier(criterion= 'entropy',max_depth= 8)
bag = BaggingClassifier(base_estimator= clf,n_estimators= 100,max_features= 0.7,n_jobs= -1)
```

圖(十) 最佳模型參數

```
              precision    recall  f1-score   support

           0       0.86      0.98      0.92      1267
           1       0.84      0.39      0.53       333

    accuracy                           0.86      1600
   macro avg       0.85      0.69      0.72      1600
weighted avg       0.86      0.86      0.84      1600

0.8575
```

圖(十一) 最佳預測結果的模型測試

	Accuracy	Precision	FScore	Total
最佳模型	0.8875	0.8333	0.6400	0.7869

RandomForestClassifier

RandomForestClassifier 利用 GridSearchCV 調出的參數，其單純測試 RandomForestClassifier 和加上 BaggingClassifier(Bagging 參數同最佳模型)後的上傳結果如下表:

	Accuracy	Precision	FScore
RF	0.87	0.778	0.57
RF+bagging	0.875	0.8	0.59

GradientBoosting

GradientBoosting 利用 GridSearchCV 調出的參數，其模型上傳結果如下表:

	Accuracy	Precision	FScore
GB	0.8725	0.7241379310344828	0.6222222222222222

ExtraTreesClassifier

ExtraTreesClassifier 利用 GridSearchCV 調出的參數，測試 ExtraTreesClassifier 加上

BaggingClassifier(Bagging 參數同最佳模型)、將 ExtraTreesClassifier 多加 max_features=8 參數以及隨機調配一套參數後的上傳結果如下表:

	Accuracy	Precision	FScore
ExtraTrees+bagging	0.845	0.857	0.367
ExtraTrees(max_features=8)+bagging	0.875	0.846	0.568
ExtraTrees (criterion= 'entropy',max_depth= 9,n_estimators=80,max_features=1)	0.8175	1.0	0.0987

CatBoostClassifier

在上網搜尋的過程中注意到 CatBoostClassifier 是比 XGBoost 和 LightGBM 更加優化的 Boosting 方法，因此我們也針對 CatBoostClassifier 進行了模型測試。以下是使用 GridSearchCV 調配出的 CatBoostClassifier 參數模型上傳結果:

	Accuracy	Precision	FScore
CatBoostClassifier	0.87	0.711864406779661	0.6176470588235294

另外因為 CatBoostClassifier 其一優點是可以不必預先處理文字型的資料，因此我們也測試了不轉換 Geography、Gender 資料，直接使用 CatBoostClassifiercat_features 的參數功能進行欄位轉換，結果如下:

	Accuracy	Precision	FScore
CatBoostClassifier	0.8675	0.8157894736842105	0.5391304347826087

VotingClassifier

VotingClassifier 我們嘗試了多種不同的組合，以下挑選其中幾項說明。

包括使用各模型透過 GridSearchCV 調配出的參數再整合一起，以及兩種 ExtraTrees+bagging 組合，以下為上傳結果:

	Accuracy	Precision	FScore
XGB+RF+最佳模型	0.8675	0.74	0.5826771653543307
前兩套 ExtraTrees 組合	0.87	0.8571428571428571	0.5357142857142857

另外因為上述 ExtraTreesClassifier 的第三套模型在 Precision 的表現為 1.0，因此我們嘗試將該模型與我們的最佳模型結合，希望能藉此提高 Precision 的表現，以下為上傳結果。

	Accuracy	Precision	FScore
ExtraTrees 第三套模型+最佳模型	0.8575	0.9166666666666666	0.4356435643564356

根據該上傳結果，確實有提升 Precision 的表現，但相對 Accuracy 及 FScore 的表現又下降太多，因此無法超越最佳模型的紀錄。

除了以上所呈現的表格外，我們還有測試許多不同的模型，而部分模型因在自行測試的階段表現結果就不理想，因此沒有上傳到網站上。在模型訓練及上載的過程中發現要如何找到三個預測指標的平衡是比較困難的部分，若提升 Precision 的表現，則相對另外兩個指標的表現又會下降。而我們最佳模型其內參數是透過人工自行隨機挑選測試而得，往後再透過 GridSearchCV 調配其他模型的參數反而都無法得到能超越該模型的數據，或者加上 RFE 的方法試圖篩選特徵也沒有發揮預期的功效。
另外上網查詢 CatBoostClassifier 時對該模型抱有很大的期待，因為網站上有寫到近期 kaggle 上多項比賽排名靠前的隊伍都使用到了 CatBoostClassifier，但經過我們的測試後仍然沒有發揮我們預期的功效。
但是在我們的最佳模型測試中有注意到 bagging 確實對提升模型預測效果有非常顯著的提升，驗證課堂中老師所整理比賽中的常勝軍幾乎都會使用到 ensemble 的手法。

6.感想與心得

陳泓吟:

雖然在這次的作業之前，我已經有學過一些機器學習的基礎知識，但透過這次的競賽實作我又學到了更多以前不知道的資料處理方法及 scikit-learn model，也學會如何利用套件調整參數篩選特徵。但比較可惜的是可能因為還沒有非常熟悉這些方法，所以這次的作業中雖然嘗試了許多不同的參數調配、特徵選擇，但始終沒有達到預期中更好的預測結果。這個部分可能就有待往後更多的經驗累積，或許在以後自行參賽或碰到相關機器學習問題時能做得更好。另外作業透過競賽的方式也讓我更有動力主動學習各種課堂上沒有提到的模型，並且不斷的嘗試。希望透過這次的經驗可以讓我們在這學期的期末專案能有更好的表現。

陳宜姍:

這次競賽讓我們自己去搜尋、了解怎麼運用 scikit-learn，還要學會自己去查各個參數的意義，並且不斷嘗試調整參數的結果。很感謝組員間互相分享各自的作法，讓我從中獲得一些啟發，從而讓預測效果更好。我覺得最困難的地方是自己的思維像是被限制住，只知道運用兩種分類模型，而且只能靠著改參數試效果，似乎過了很久都沒有太大進展、很難找到突破點。不過透過這次競賽，我確實有更了解機器學習在做什麼，並且透過上傳成績與同學競爭，增加不少動力與成就感。很感謝這次經驗，讓我知道自己的能力到哪以及有哪些不足的地方，也相信這些經驗會讓我們在期末專題中進行更順利。

陳凱騫:

在這次的功課中，我原先以為只要將模型套入並且避免過度配飾就好了，但是一開始測試出來的結果卻十分不好，讓我知道其實機器學習十分注重資料前處理的部分，因此我在資料前處理也另外在上網查詢了許多資料，也是我在這次作業中花費最多時間的部分，而我也在這次競賽中自己上網找了許多不同的模型來嘗試，讓我知道了許多模型的運作及對於不同資料的適合度，此外在這次的競賽中，也讓我知道了原來這種

程式的競賽是如此進行的，使我也對於這些競賽產生了些許興趣，未來如果有機會應該也會再參加不同類型的競賽。