ABC Competition Report

隊伍名稱

熬夜寫 Code 來杯 Java

組員

F74114037 江曉明

F74116275 陳柏淮

F74111071 楊承翰

壹、 競賽介紹

ABC Competition(A general model for Binary Classification)

如名稱所述，本比賽的目標是建構一種具有高泛化能力的二元分類模型，期望該模型能在多種二元分類問題上表現良好。

本競賽總共有 49 個資料集，需要我們針對這些資料集，產生對應的模型、預測給定的測試資料，並上傳預測結果。

由於某些資料集的類別標籤（如 Dataset 4）存在嚴重的不平衡，因此本競賽選擇使用 AUC（Area Under the Curve）作為評分標準，以更公平地評估模型在不平衡資料集上的表現。

貳、 最終模型建構

一、 資料前處理與特徵工程：

無任何資料前處理

二、 模型選擇：

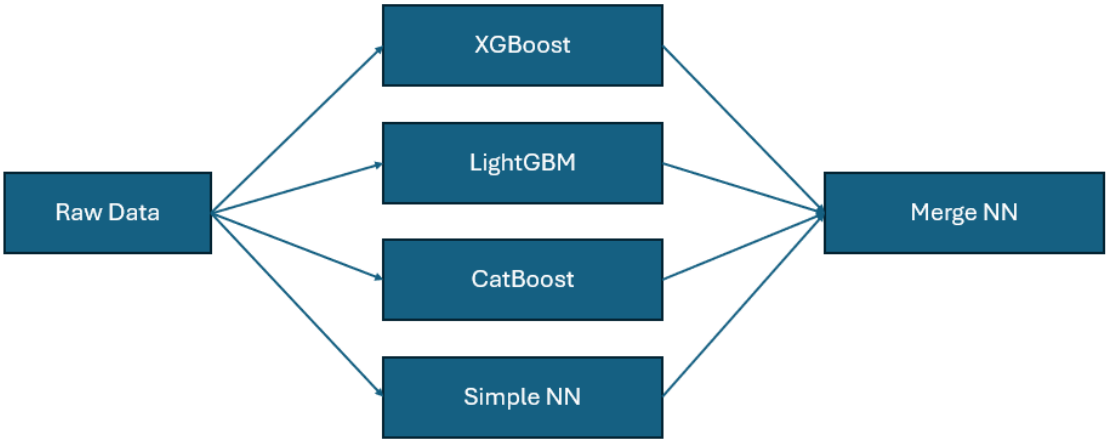
使用到 XGBoost、lightGBM、catBoost 以及 Neural Network。

1. 將原始資料（無任何前處理）作為輸入，類別標

籤作為輸出，訓練 XGBoost、LightGBM、CatBoost 以及 Neural Network（在此稱為 Simple NN）。

2. 將四個模型的輸出當作輸入，類別標籤當作輸出，再訓練一個 Neural Network（也就是 Ensemble model，但在此稱為 Merge NN）。
3. Merge NN 的結果即為最終預測結果。

流程圖如下



三、 模型簡述

XGBoost:

1. 使用 RandomizedSearchCV 以及 StratifiedKFold 找出最佳超參數。

2. 找出最佳超參數後，再利用全部的 train data 來訓練模型。(註 1)

   詳細參數請參考 ming 資料夾下的 XGBoost.py

LightGBM:

1. 使用 GridSearchCV 找出最佳超參數
2. 找出最佳超參數後，再利用全部的 train data 來訓練模型。(註 1)

   詳細參數請參考 ming 資料夾下的 lightGBM.py

CatBoost:

1. 使用 RandomizedSearchCV 以及 StratifiedKFold 找出最佳超參數。
2. 找出最佳超參數後，再利用全部的 train data 來訓練模型。(註 1)
3. 將 categoric features 當作 numeric features 處理。(註 2)

   詳細參數請參考 ming 資料夾下的 catBoost.py

SimpleNN:

```python
class SimpleNN(nn.Module):
    def __init__(self, input_size):
        super(SimpleNN, self).__init__()
        self.fc1 = nn.Linear(input_size, 64)
        self.dropout1 = nn.Dropout(0.2)
        self.fc2 = nn.Linear(64, 32)
        self.dropout2 = nn.Dropout(0.2)
        self.fc3 = nn.Linear(32, 1)

    def forward(self, x):
        x = F.relu(self.fc1(x))
        x = self.dropout1(x)
        x = F.relu(self.fc2(x))
        x = self.dropout2(x)
        x = torch.sigmoid(self.fc3(x))
        return x
```

損失函數：Binary Cross Entropy (BCELoss)

優化器：Adam

詳細參數請參考 ming 資料夾下的 simple_NN.py

MergeNN (Ensemble model):

```python
class MergeNN(nn.Module):
    def __init__(self, input_size):
        super(MergeNN, self).__init__()
        self.fc1 = nn.Linear(input_size, 64)
        self.fc2 = nn.Linear(64, 1)

    def forward(self, x):
        x = F.relu(self.fc1(x))
        x = torch.sigmoid(self.fc2(x))
        return x
```

損失函數：Binary Cross Entropy (BCELoss)

優化器：Adam

詳細參數請參考 ming 資料夾下的 merge_NN.py

四、註釋：

1. 由於一般都會將資料再分成訓練集與驗證集，會導致總有一部份資料無法用於訓練模型，但是只要先找出最佳超參數（即已確認該超參數組合能讓模型有良好的表現），便可以利用全部的訓練資料重新訓練模型，以提高在測試集的表現。
2. CatBoost 本身可以透過創建 CatBoostClassifier Object 時，傳入參數 cat_features 來讓其自行處理類別特徵。

   然而，測試過後，發現使用該參數處理類別特徵並無法讓分數提高，且訓練模型時間大幅增加。

   因此決定將原先類別特徵當作數值特徵處理。

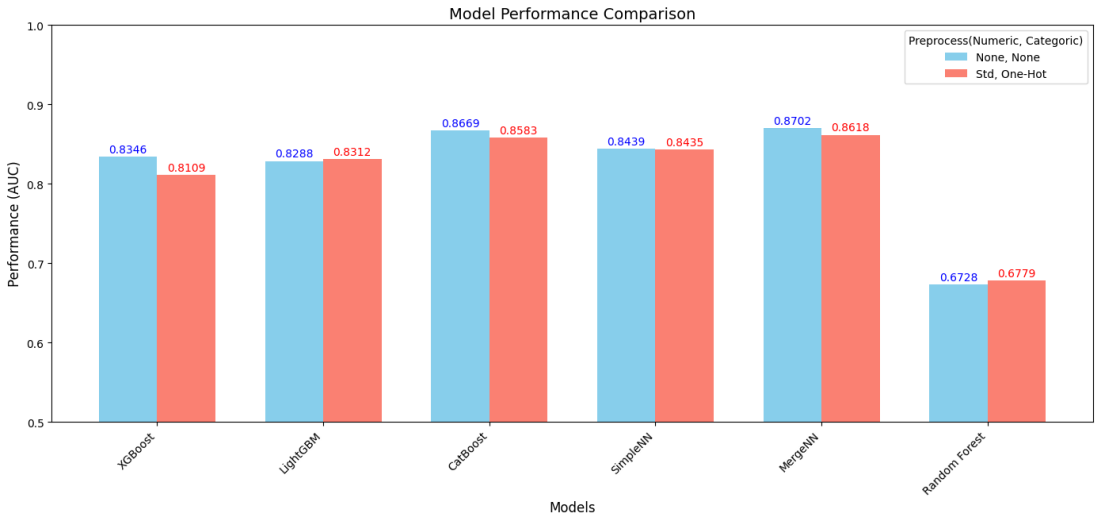
   圖示：

```
base_model = CatBoostClassifier(
        eval_metric="AUC",
        random_seed=RANDOM_SEED,
        verbose = 0,
        # cat_features = list(range(len(numeric_features), X_train.shape[1])),
        iterations=100
    )
```

參、 資料分析

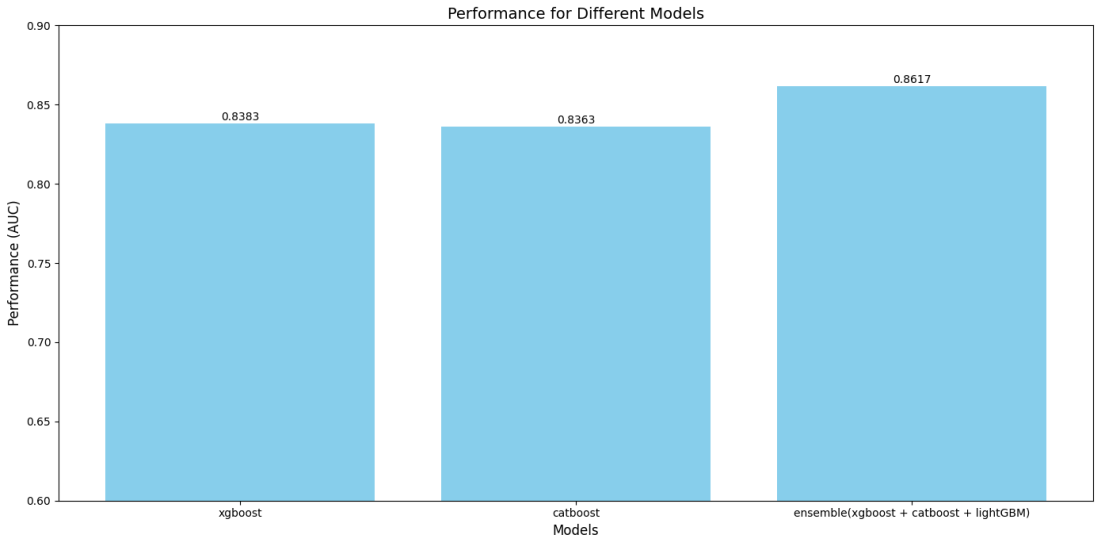
一、 不同模型的效果

江曉明的測試結果



圖（一）

楊承翰的測試結果



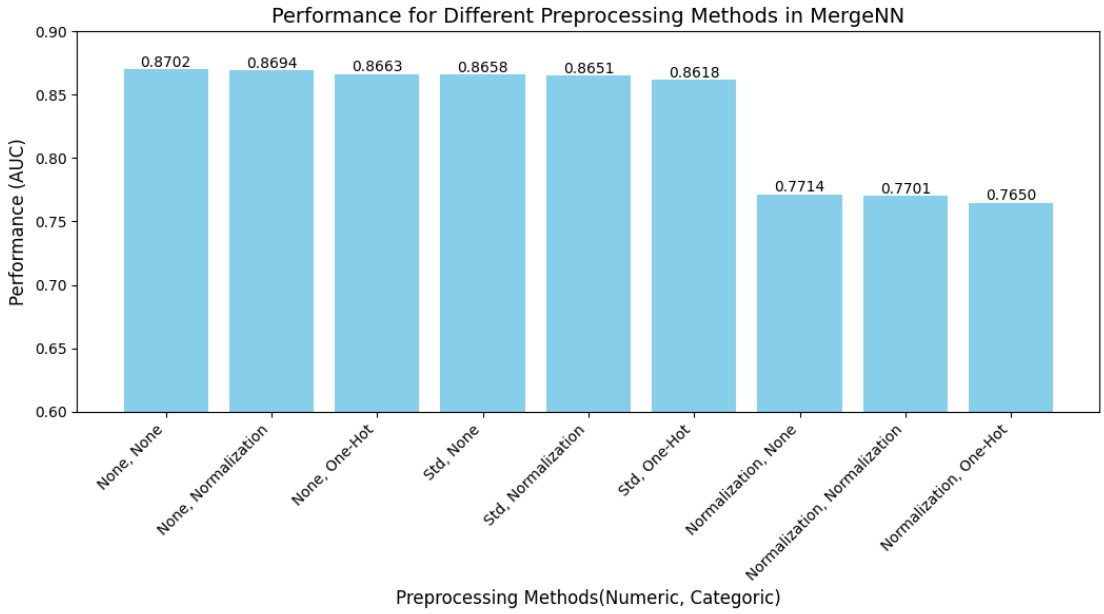
圖(二)

從圖(一)和圖(二)可以得知

- 對於大部分模型，不做任何資料前處理的模型比起做了資料前處理（標準化與獨熱編碼）的模型，效果更好。
- 模型效果排名：

  MergeNN > CatBoost > SimpleNN > XGBoost > LightGBM > Random Forest
- Ensemble Model 的效果較單一模型好。

二、 Ensemble Model 在不同資料前處理的效果



圖(三)

從圖(三)可以得知

- 數值特徵前處理效果：無前處理 > 標準化 > 正規化。

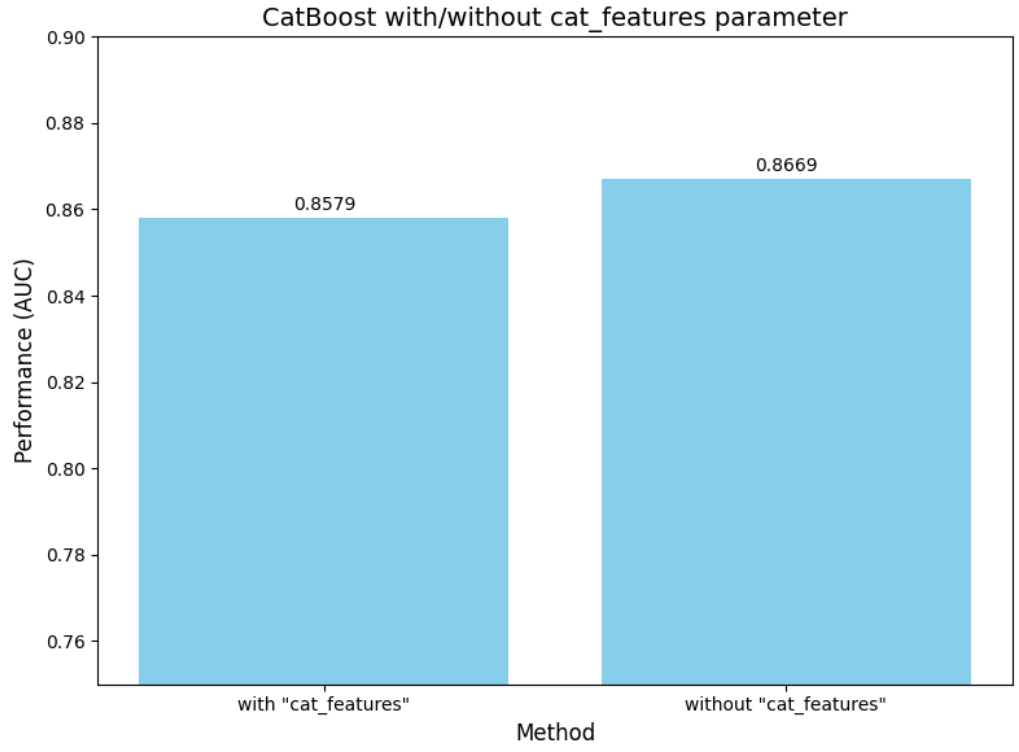
- 類別特徵前處理效果：無前處理 > 正規化 > 獨熱編碼。

註：

標準化：透過 $z = \frac{x-\mu}{\sigma}$ *其中* $\mu =$ *平均數*、$\sigma =$ *標準差* 公式，對資料做轉換。

正規化：透過 $x' = \frac{x-\min(x)}{\max(x)-\min(x)}$ 公式，對資料做轉換。

三、 CatBoost 使用參數 cat_features 的效果



圖(四)

從圖(四)可以得知

- 使用 cat_features 參數效果並沒有比較好。

肆、 總結

本競賽的目標是建立具有高泛化能力的二元分類模型，並在多個資料集上表現出色。經過模型建構與測試，得出以下主要結論：

一、資料前處理的影響：

對於大部分模型而言，無前處理的效果反而比進行資料前處理（標準化與獨熱編碼）更好。

在數值特徵的前處理上，無前處理的效果最佳，其次是標準化，再來是正規化。

在類別特徵的前處理上，無前處理效果最佳，其次是正規化，再來是獨熱編碼。

這些結果顯示，對於特定的資料集與模型，過度的資料轉換可能會帶來負面影響，反而直接使用原始資料能獲得更好的表現。

二、模型表現：

合併不同模型的預測結果（即使用 MergeNN）顯示

出集成方法在多個資料集上能提供更穩定且更具泛化能力的表現。

三、模型訓練的策略：

XGBoost、LightGBM 和 CatBoost 模型皆先使用交叉驗證和超參數搜尋來選出最佳超參數，並最終用全部訓練資料重新訓練模型。這樣的策略有助於利用所有資料來提高模型在測試集上的預測能力。

在處理類別特徵方面，儘管 CatBoost 提供了 cat_features 參數來自動處理類別特徵，但測試結果顯示，這對模型效果的提升幫助不大，反而增加了訓練時間。

因此，將類別特徵視為數值特徵進行處理能達到較好的效果。

總結來說，本組在這次競賽中的最佳方法為，利用多種不同模型進行訓練，並將其結果進行集成（如 MergeNN）。此外，避免過多的資料前處理可能能提高模型表現。

伍、 分工