Competition_1
成果報告

組別：等一下吃什麼！
組員：統計學系116級 蔡馨漩、周子喬、張詠嵐
指導教授：國立成功大學資訊工程學系 李政德 教授
日期：113年12月2日

目錄

一、專案介紹 (Introduction)..... 3
二、資料前處理 (Data Preprocessing)..... 3
三、模型建構與訓練 (Model Building & Training)..... 3
四、評估指標 (Evaluation Metrics)..... 3
五、結果分析與討論 (Results & Discussion)..... 3
六、結論 (Conclusion)..... 3
七、附錄 (Appendix)..... 3

一、專案介紹 (Introduction)

1. 背景與問題

我們將透過老師提供的資料集進行分析、建模，並產生對應的預測結果。最終分數會是模型在每個資料集上的預測表現之加權平均因此設計的資料處理流程與模型必須具有通用性，方能在大多數資料集上具有好的表現。

2. 資料集描述

本次競賽所使用的資料集為49個二分類任務之小資料集(資料筆數小於10000)，資料集內有X_train, X_test, y_train, y_predict四個csv檔案 y_predict為我們需要透過模型預測產生的。

3. 競賽目標

透過嘗試不同的二元分類方法，將資料及套入設計的模型中進行訓練，持續優化模型以達到AUC(Area under the Curve, 0<AUC<1)趨近於1，以驗證模型的正確率。

4. 使用模型與結果

本組在競賽過程嘗試了RandomForest、XGBoost、LogisticRegression三種模型，最後採用LogisticRegression的結果作為最終預測模型上傳。
最終結果：private score: AUC=0.798498

二、資料前處理 (Data Preprocessing)

1. 資料檢查

透過函數 `check_data_quality(df, dataset_name)`，檢查數據，包括：

1. 缺失值檢查：使用 `isnull().sum()` 來統計每列中的缺失值數量。
2. 重複行檢查：使用 `duplicated().sum()` 檢查是否存在重複的行。
3. 基本統計信息：使用 `describe()` 生成數值型特徵的統計摘要。
4. 異常值檢查：利用 IQR（四分位距）方法檢測可能的異常值。

2. 缺失值處理

函數 `handle_missing_values(df)` 針對不同類型的特徵進行缺失值填充：

1. 數值型特徵：缺失值用中位數填充（`fillna(median)`）。
2. 類別型特徵：缺失值用眾數填充（`fillna(mode)`）。

3. 特徵縮放

函數 `scale_features(X_train, X_test)` 進行了數據標準化，主要針對數值型特徵進行縮放：

a. 使用了 標準化（Standardization）：通過 `StandardScaler()` 將數據轉換為零均值和單位標準差的分佈。
b. 提供了選擇其他縮放方法的框架：
    i. MinMaxScaler()：將數據縮放到 [0, 1] 區間。
    ii. RobustScaler()：針對異常值的穩健縮放。

4. 資料讀取

程式載入資料，並將其分為多個資料集：

a. `X_train`, `y_train`, `X_test` 分別從每個資料夾中讀取對應的 CSV 檔案。
b. 使用 `pandas.read_csv()` 方法將資料載入為 DataFrame。

三、模型建構與訓練 (Model Building & Training)

1. RandomForest

a. 簡介

Random Forest是一種基於決策樹的集成學習方法。它通過構建多棵Decision Tree，並對其結果進行投票來提高模型的準確性和穩定性。隨機森林在每棵樹的訓練過程中，隨機選擇部分樣本和部分特徵（Bootstrap），與單一決策樹相比，隨機森林能有效減少過擬合的風險。

b. 套件使用

```
from sklearn.ensemble import RandomForestClassifier
```

c. 模型訓練與調整參數

i. `n_estimators=3000`
ii. `max_depth=None`
iii. `class_weight='balanced'`

d. 模型預測結果

private score: AUC=0.812619

e. 部分程式碼

```
#rf模型建構
rf_model = RandomForestClassifier(
    n_estimators=3000,
    max_depth=None,
    random_state=42,
    class_weight='balanced')
rf_model.fit(X_train_combined, y_train_combined)
```

2. XGBoost

a. 簡介

XGBoost是一種基於Gradient Boosted Decision Tree的機器學習演算法。它的核心基於樹集成模型，每次增量訓練新增一棵樹以修正前一步的預測誤差，逐步優化模型的目標函數。

b. 套件使用

```
from xgboost import XGBClassifier
```

c. 模型訓練與調整參數

使用XGBoost作為模型，超參數經過多次調整，最佳結果如下：

i. n_estimators=2000：設定較大的決策樹數量，讓模型有足夠的學習能力。
ii. max_depth=None：設定為None表示不限制樹的深度，允許樹完全擬合數據的結構。
iii. 當數據特徵和樣本量較多時，更深的樹可以幫助建立更細緻的模型。
iv. learning_rate=0.04：設置小的學習率可以減少每次更新對模型的影響，通常配合數量較多的決策樹一起使用以達到更高的準確性。
v. colsample_bytree=0.9：每棵樹將隨機使用其中的90%特徵進行訓練。
vi. colsample_bylevel=0.9：每層分裂時，在90%的特徵中進行選擇。
vii. subsample=0.9：每棵樹隨機選擇資料集內90%的數據進行訓練。
viii. scale_pos_weight=len(y_train_combined[y_train_combined == 0]) / len(y_train_combined[y_train_combined == 1])：調整正負類別樣本的權重，＝負類樣本數 / 正類樣本數

d. 模型預測結果

private score:AUC=0.821101

e. 部分程式碼

```
#建構xgboost模型
xgb_model = XGBClassifier(n_estimators=2000,
                          max_depth=None,
                          learning_rate=0.04,
                          colsample_bytree=0.9,
                          colsample_bylevel=0.9,
                          subsample=0.9,
                          random_state=42,
                          scale_pos_weight=len(y_train_combined[y_train_combined == 0]) / len(y_train_combined[y_train_combined == 1]))
xgb_model.fit(X_train_combined, y_train_combined)
```

3. Logistics Regression

a. 簡介

Logistic Regression 是一種用於分類問題的統計模型，特別適合二元分類（例如，是/否、正/負等）。其核心思想是利用邏輯函數（sigmoid 或 logit 函數）將線性迴歸的輸出轉換為機率值，從而對樣本進行分類。

b. 套件使用

```
from sklearn.linear_model import LogisticRegression
```

c. 模型訓練與調整參數

用訓練資料（`tmp_X_train` 和 `tmp_y_train`）擬合 Logistic Regression 模型。

model.fit(tmp_X_train, tmp_y_train.squeeze())

- `C=1.0`: 正則化參數，控制模型的複雜度，值越小正則化越強。
- `penalty='l2'`: 使用 L2 正則化來防止過擬合。
- `solver='lbfgs'`: 最優化算法，用於優化對數似然函數。
- `class_weight='balanced'`: 根據數據集類別分布自動調整權重。

d. 模型預測結果

private score:AUC=0.839923

e. 部分程式碼

```
model = LogisticRegression(
    random_state=42,
    max_iter=1000,
    C=1.0,  # 調整這個值：較小的值增加正則化強度
    penalty='l2',  # 可以改用 'l1' 進行特徵選擇
    solver='lbfgs',  # 可以嘗試不同的優化器
    class_weight='balanced'
)
```

四、評估指標 (Evaluation Metrics)

AUC (Area Under the Curve) 評分標準：

數學定義：表示隨機選擇一個正類樣本和一個負類樣本時　，分類器將正類樣本的得分判為高於負類樣本的概率。

範圍解釋：

- AUC = 1: 完美分類，模型能正確區分所有正類和負類樣本。
- AUC = 0.5: 隨機分類器，無法從正類和負類樣本中做有效區分。
- AUC < 0.5: 模型性能比隨機分類器還差（可能是模型標籤反轉造成的）。

我們的最高分：AUC=0.839923

五、結果分析與討論 (Results & Discussion)

1. 關鍵觀察

   綜合三種我們嘗試的訓練模型，發現對於這數筆資料集來說，Logistic Regression 的二元分類資料所呈現的AUC數值最高。

2. 模型限制

   以下是分析此三種模型的使用限制與優勢：

	限制	優勢
RandomForest	1. 在特徵數目較多時可能會變得計算量較大。 2. 相比 XGBoost 和 Logistic Regression，模型結果稍弱，可能與調參不足或模型特性有關。	1. 對異常值不敏感，能有效降低過擬合風險。 2. 不需要對數據進行過多的前處理（如標準化）。
XGBoost	1. 超參數調整需要較多的時間和計算資源。 2. 低學習率可能會導致訓練時間變長。	1. 能更好地處理異常值和特徵重要性偏重的情況。 2. 支持特徵重要性排名，有助於解釋模型。
LogisticRegression	1. 可能不適用於非線性數據。 2. 所能處理的資料量不可過於旁大。	1. 邏輯回歸計算效率高，模型解釋性強。 2. 對高維數據效果良好。

六、結論 (Conclusion)

在這次的競賽中，我們總共嘗試了三種二元分類方式：
Logistic Regression 表現最佳（AUC=0.839923），可能得益於其模型的高解釋性和適當的正則化策略。
XGBoost 表現居中（AUC=0.821101），展現了其在處理複雜結構和異常值方面的優勢。
Random Forest 雖結果稍弱（AUC=0.812619），但仍具有穩定性和抗過擬合能力的優點。

七、附錄

a. 完整程式碼(github)

   https://github.com/mellamochiao/datascienceCP1.git

b. 參考連結

   i. Random Forset
      https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestClassifier.html

   ii. XGBoost github.com/dmlc/xgboost