

T-Brain 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

競賽全名為區域微氣候資料預測發電量競賽，參賽者需要透過今年度花蓮氣象站附近的 17 台太陽能發電設備以及周邊裝置收集到的數據，如風速、氣壓、溫度、濕度及日照強度等特徵，來預測測試集中指定日期的缺失發電量。

此次競賽共有 17 份資料集以及部分補充資料，每份資料包含裝置 ID、時間及當天 7:00 到 17:00 每 10 分鐘一筆的 5 項氣候特徵跟發電量，但由於設備具有不穩定性以及感測限制，加上時間序列的不完整性，這些設計都大大增加了本次競賽的難度。

參賽者需要預測官方指定共 200 天每天 9:00 到 17:00 的發電量，總計 9600 筆資料，競賽採用預測值與實際發電量的之差的絕對值和（MAE）作為評分依據，總分數越低則代表模型的預測越準確，參賽者排名越高。

貳、 資料前處理過程

一、 訓練資料整合

官方提供的資料集有分一般資料跟額外資料，由於只有部分裝置具有額外資料，需先將這些裝置的資料與原本

的資料集整合。

二、訓練資料分類

由於官方提供的資料集不利於答案的預測處理，官方有提供額外外掛程式幫助進行資料前處理，透過這個程式自行切割驗證集跟訓練集，也可以將資料的時間跟裝置 ID 特徵轉變成上傳格式使用的 Serial，並將 17 份資料集分類成早上 7:00 到 9:00 的 Incomplete Avg Data 還有需要被預測的時間段 9:00 到 17:00 的 Avg Data，這樣的好處是可以利用時間序列模型如 LSTM，藉由當天早上 2 小時的資訊預測未來的氣候特徵及發電量。

三、透過氣象局資料進行額外特徵爬取

這次競賽雖然提供了 5 項與發電量相關的氣候特徵，但因為我們認為太陽能板的發電量還會跟當天太陽的方位角、仰角等特徵相關，所以我們透過交通部中央氣象局的每日天文現象網站（註 1），獲取當天的日出和日落的時間跟方位角，以及日中天當下太陽的仰角，再加上官方提供的每個太陽能板的緯度資訊（註 2），透過這些資訊即可用程式算出當日每時每刻的太陽仰角跟方位角資訊。

註 1：交通部中央氣象署天文資料

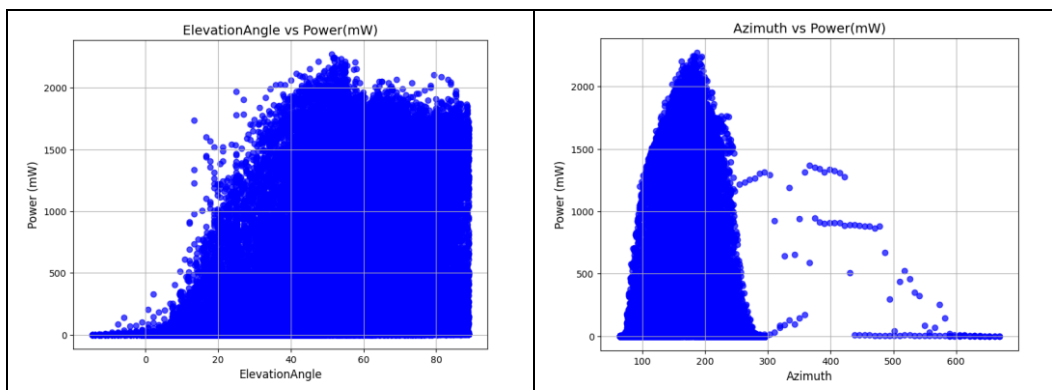
https://www.cwa.gov.tw/V8/C/K/astronomy_day.html

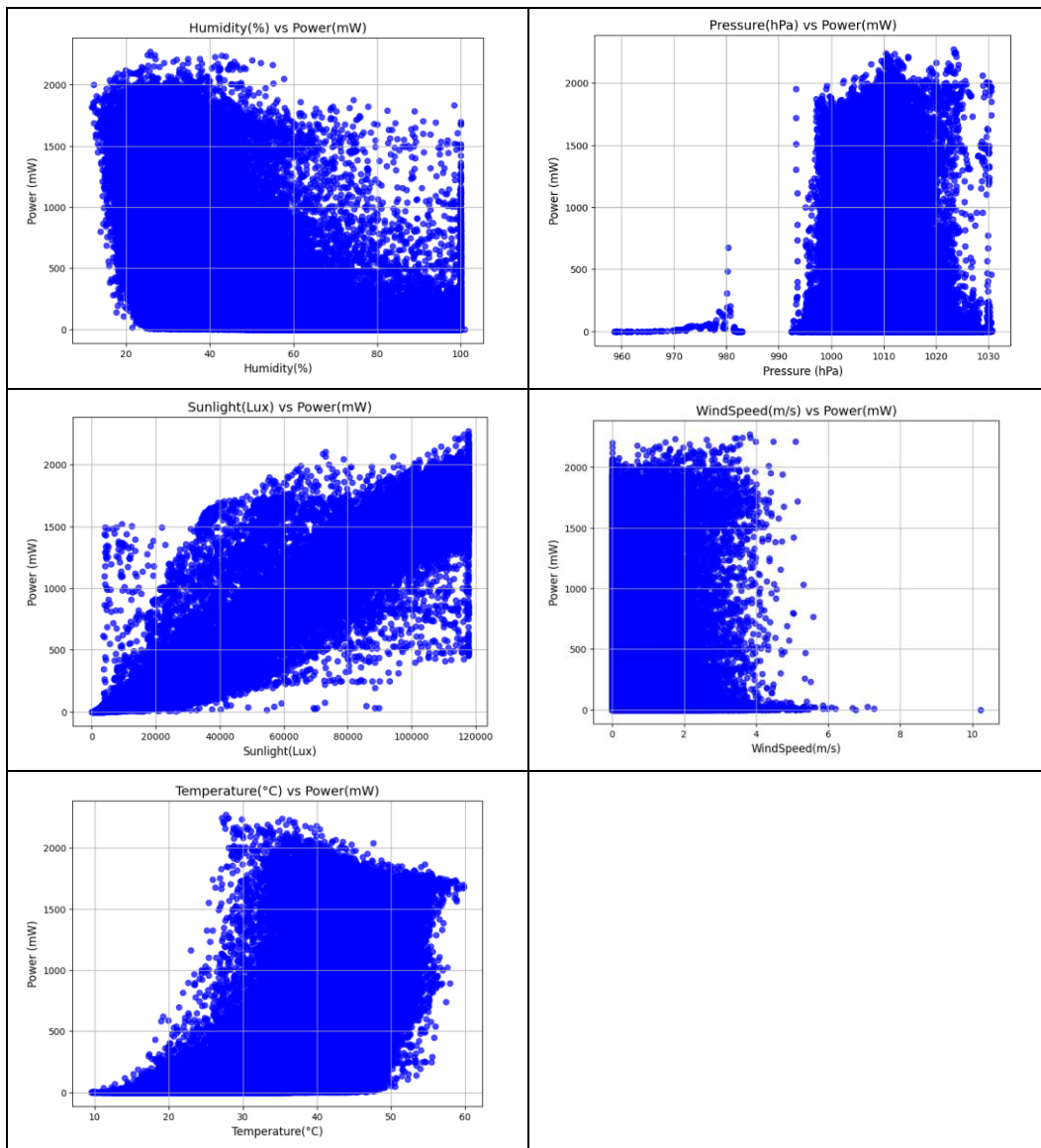
註 2：裝置 17 號官方只提供位置在花蓮氣象局附近 2 公里內，根據常理判斷，緯度可以採用花蓮氣象局的緯度。

四、資料視覺化

透過散布圖將資料進行視覺化，觀察各個特徵與目標

（發電量）的關係，Y 軸為目標，X 軸為各個特徵，包括官方提供的溼度、氣壓、光照、風速、以及每日天文現象網站爬取的方位角和仰角，可以看出除了風速和發電量的變化關係甚微，故我們將了風速以外的特徵皆納入訓練。





五、特徵前處理

將剛剛獲得的 17 份 Incomplete Avg Data 和 17 份 Avg Data

整合，整合成 3 份完整的.csv 檔，分別是只包含

Incomplete Avg Data 的 Train data、只包含 Avg Data 的

Train data 和包含兩者的 Train data。

對這 3 份資料集的特徵透過程式進行額外處理。

1、 透過 Serial 行可獨立出 Datetime、Date、Month、

Hour、Minute 和 DeviceID 等資訊。

- 2、 因為 DeviceID 屬於類別型資料，將其進行 One-Hot 編碼，只保留數值為 0 或 1。
- 3、 我們認為過去的發電量也能作為特徵輸入模型用於訓練，所以我們用每一筆資料過去兩小時內每 10 分鐘一筆的發電量創造滯後特徵共 12 筆，並將創造滯後特徵後產生缺失的列移除。
- 4、 其他數值型資料將風速移除，保留氣壓、溫度、濕度、光照，並且不進行標準化(使用 LSTM 模型時需要進行正則化，因為 LSTM 模型對數字大小較為敏感)，因為測試後發現直接使用原始資料訓練能保留更完整的特徵，進而獲得更高的準確度。

參、 缺失特徵預測

我們認為這次競賽的最大難點在於利用時間序列不夠完整的資料集，預測資料缺失當天的氣候特徵，為了解決這個問題，我們試過以下幾種方法：

一、 LSTM 時間序列模型加回歸模型（註 6）

此種方式即是 Sample Code 採用的方法，將 Avg Data 作為 Train Data 傳入，而後利用需預測的缺失值當天早上

的 Incomplete Avg Data 來預測整天的數據，這邊僅用於預測特徵，並修改 LSTM 的 activation 參數為'relu'，使其不會預測出負值。

缺點：因為需要用到預測值當作下一個預測值的特徵，可能會導致錯誤累加的問題。

二、 回歸模型（註 6）

使用回歸模型產生的預測特徵值。

缺點：回歸模型對於時間序列的數據較難捕抓相關性，簡單的回歸模型較難處理時間序列這類具有週期性且非線性的關係，使用預測值作為輸入，也會有錯誤累加的問題。

三、 平均特徵

透過取得要預測當天附近的氣候特徵，例如前後各 2 天、5 天或 10 天的相同時間段的值進行平均，來填補缺失的當天特徵。

缺點：若短期內的氣候幅度變化劇烈，平均值較難反映當下真實的氣候特徵，另外，被要求要預測的時間段附近通常具有較大的資料缺失，甚至沒有過去或未來的資料，導致平均值不平衡。

註 6：此類模型也可以直接預測發電量，這邊僅用於預測氣候特徵。

肆、 模型訓練過程

使用自製的 Train(AVG)資料（註 7）加上天文網站爬取的特徵（方位角和仰角）進行訓練，透過應預測日期前後 5 天的資料預測當天各個時段的發電量，另外，因為各個地點(裝置 ID)結果差距較大，因此我們選擇針對 17 個地點個別訓練模型。模型採用 Ensemble Model，結合 XGBoost 和 Regression Neural Network 的預測結果作為輸入，提高預測的準確性和穩定性。

註 7：由於官方提供的 AVG 以及 IncompleteAVG 並沒有包含所有資料，再加上後來官方又提供額外資料，因此需要再自行再做整合。詳細可以參考 Training_Data\ming 資料夾中的 Readme.md。

一、 XGBoost

使用 RandomizedSearchCV 以及交叉驗證找出最佳超參數，並使用 XGBoost 提供的回歸模型，設定目標函數為平均絕對誤差（MAE）。


```

param_dist = {
    'learning_rate': uniform(0.01, 0.3),
    'max_depth': randint(3, 10),
    'n_estimators': randint(100, 501),
    'subsample': uniform(0.6, 0.4),
    'colsample_bytree': uniform(0.6, 0.4),
    'reg_alpha': uniform(0, 1),
    'reg_lambda': uniform(1, 2),
}

xgb_model = xgb.XGBRegressor(
    objective='reg:absoluteerror',
    random_state=random_seed
)

```

(XGBoost 模型架構如上圖所示)

二、 Regression Neural Network:

使用平均絕對誤差 (L1Loss) 作為損失函數，以及

Adam 優化器，並添加 weight_decay 防止過擬合。

```

class RegressionModel(nn.Module):
    def __init__(self, input_dim, input_features):
        super(RegressionModel, self).__init__()
        self.fc = nn.Sequential(
            nn.Linear(input_dim, 1024),
            nn.ReLU(),
            nn.Linear(1024, 512),
            nn.ReLU(),
            nn.Linear(512, 512),
            nn.ReLU(),
            nn.Linear(512, 128),
            nn.ReLU(),
            nn.Linear(128, 64),
            nn.ReLU(),
            nn.Linear(64, 1)
        )
        self.input_features = input_features

    def forward(self, x):
        return self.fc(x)

```

(Regression NN 模型架構如上圖所示)

三、 Random Forest

使用 RandomizedSearchCV 以及交叉驗證找出最佳超

參數，並使用 Random Forest 提供的回歸模型，設定

目標函數為平均絕對誤差（MAE）。

```
param_dist = {  
    'n_estimators': randint(100, 501),  
    'max_depth': randint(3, 20),  
    'min_samples_split': randint(2, 10),  
    'min_samples_leaf': randint(1, 5),  
    'max_features': [0.6, 0.7, 0.8, 0.9, 1.0],  
    'bootstrap': [True, False],  
}
```

(Random Forest 模型架構如上圖所示)

四、 Ensemble Model

將自製的 AVG 資料集加上 XGBoost 和 Regression

Neural Network 的預測結果作為輸入，再訓練一個

Ensemble model（Neural Network）預測最終答案。

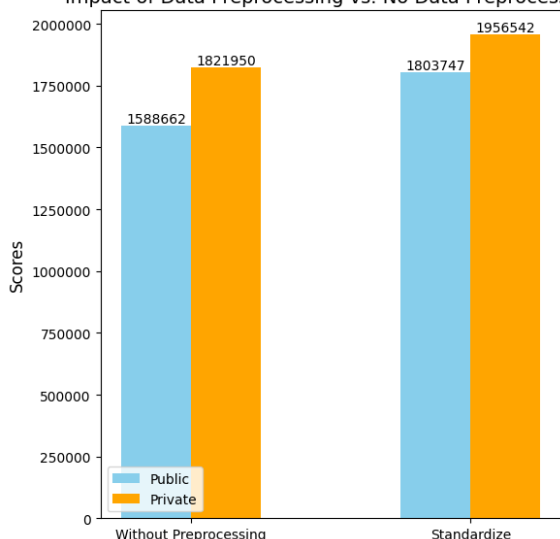
```
class EnsembleRegressionModel(nn.Module):  
    def __init__(self, input_dim, input_features):  
        super(EnsembleRegressionModel, self).__init__()  
        self.fc = nn.Sequential(  
            nn.Linear(input_dim, 1024),  
            nn.ReLU(),  
            nn.Linear(1024, 512),  
            nn.ReLU(),  
            nn.Linear(512, 512),  
            nn.ReLU(),  
            nn.Linear(512, 64),  
            nn.ReLU(),  
            nn.Linear(64, 1)  
        )  
        self.input_features = input_features  
  
    def forward(self, x):  
        return self.fc(x)
```

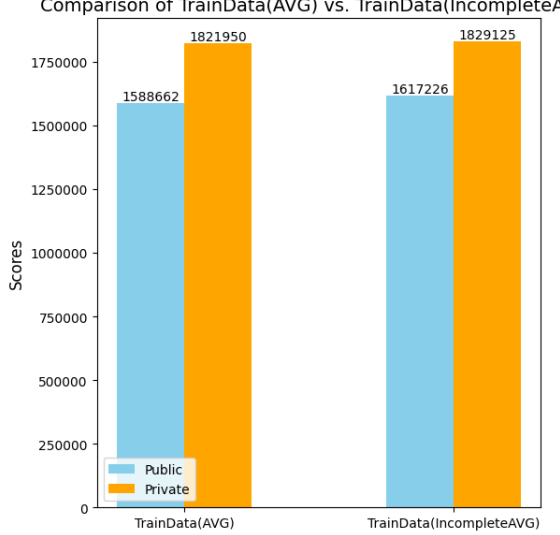
(Ensemble NN 模型架構如上圖所示)

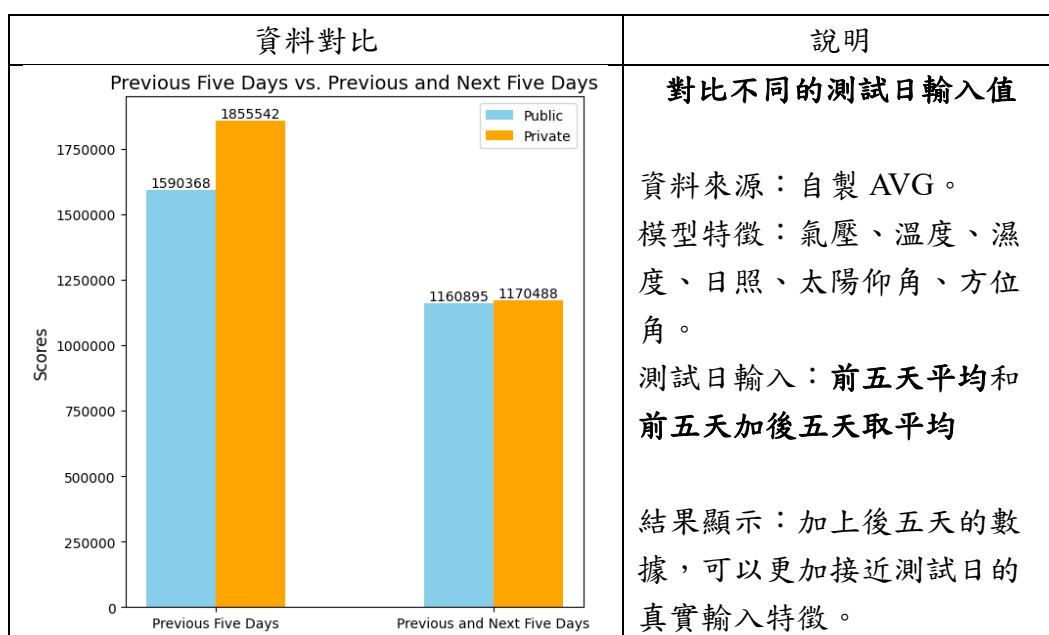
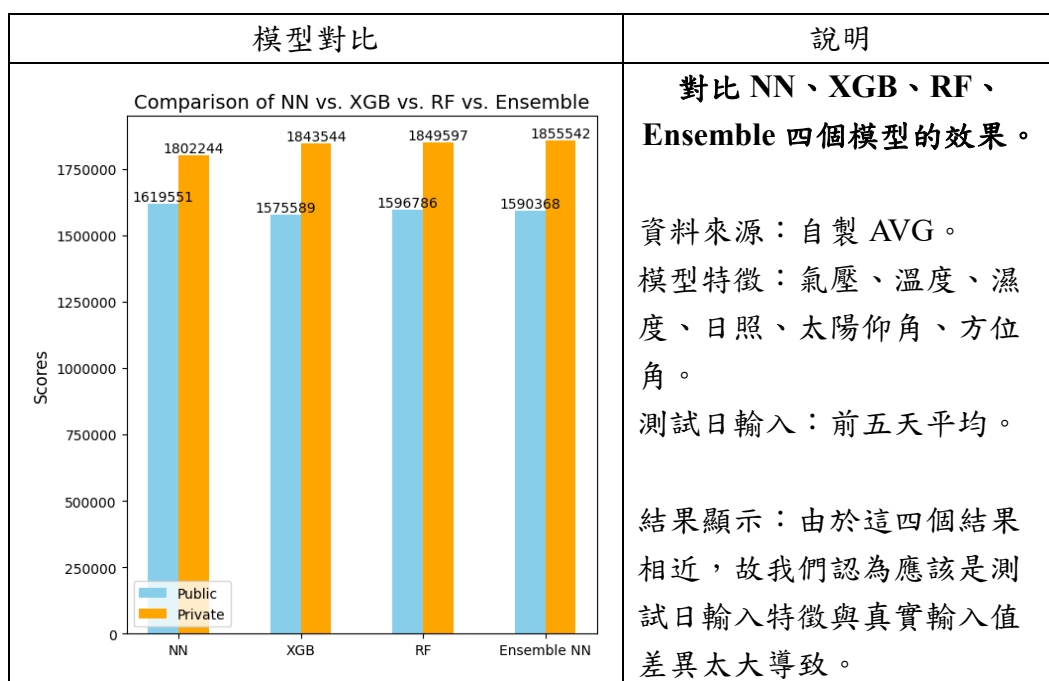
使用平均絕對誤差（L1Loss）作為損失函數，以及

Adam 優化器，並添加 weight_decay 防止過擬合。

伍、模型成果評估

資料前處理對比	說明									
<p>Impact of Data Preprocessing vs. No Data Preprocessing</p>  <table><thead><tr><th>Preprocessing</th><th>Public</th><th>Private</th></tr></thead><tbody><tr><td>Without Preprocessing</td><td>1588662</td><td>1821950</td></tr><tr><td>Standardize</td><td>1803747</td><td>1956542</td></tr></tbody></table>	Preprocessing	Public	Private	Without Preprocessing	1588662	1821950	Standardize	1803747	1956542	<p>對比標準化的有無</p> <p>資料來源：自製 AVG。</p> <p>模型特徵：氣壓、溫度、濕度、日照。</p> <p>測試日輸入：將該測試日的前五天做平均。</p> <p>結果顯示：不進行資料標準化前處理有更好的效果（誤差較低）。</p>
Preprocessing	Public	Private								
Without Preprocessing	1588662	1821950								
Standardize	1803747	1956542								

不同資料集對比	說明									
<p>Comparison of TrainData(AVG) vs. TrainData(IncompleteAVG)</p>  <table><thead><tr><th>Train Data</th><th>Public</th><th>Private</th></tr></thead><tbody><tr><td>TrainData(AVG)</td><td>1588662</td><td>1821950</td></tr><tr><td>TrainData(IncompleteAVG)</td><td>1617226</td><td>1829125</td></tr></tbody></table>	Train Data	Public	Private	TrainData(AVG)	1588662	1821950	TrainData(IncompleteAVG)	1617226	1829125	<p>對比自製 IncompleteAVG 和 AVG 資料集的效果</p> <p>資料來源：自製 AVG 以及 IncompleteAVG 的資料。</p> <p>模型特徵：氣壓、溫度、濕度、日照。</p> <p>測試日輸入：將該測試日的前五天做平均。</p> <p>結果顯示：使用 AVG 有更好的效果。</p>
Train Data	Public	Private								
TrainData(AVG)	1588662	1821950								
TrainData(IncompleteAVG)	1617226	1829125								



陸、 競賽成果

Public Leaderboard:

104	TEAM_6982	3	30	1160895.4	11/28/2024 8:12:27 PM
------------	-----------	---	----	-----------	--------------------------

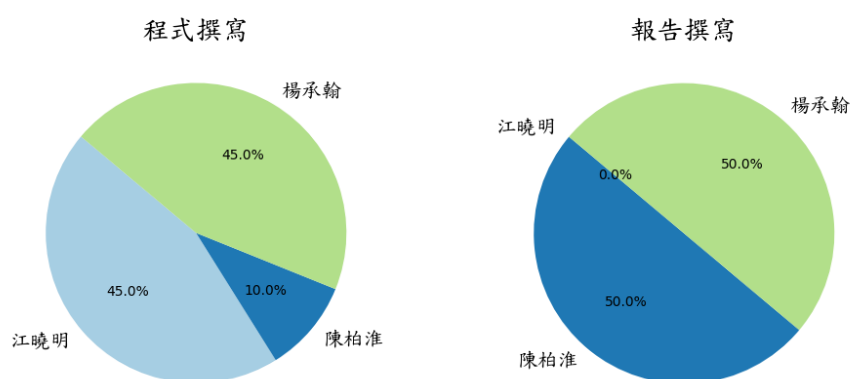
Private Leaderboard:

91	TEAM_6982	3	30	1170488.	11/28/2024 8:12:27 PM
-----------	-----------	---	----	----------	--------------------------

雖然這次比賽我們未能找到較好的特徵值預測方法，導致最終競賽的成果有些不盡理想，但從 Private Leaderboard 的表現來看，我們的模型具有較強的泛化能力，潛在的 overfitting 問題較小，兩種排行的分數並沒有太大差距，進而使排名上升。

柒、 其他

分工



GitHub Repository

<https://github.com/MingMinNa/T-Brain-Competition>