資料探勘 期末專題小組整合報告

台灣稻米產量相關分析與預測

第十組：李家維、陳奕廷、李承恩

一、專題介紹

1.題目介紹：

以**「台灣稻米產量」**為中心主軸，進行台灣稻米產量相關分析，藉由找尋相關 features 進行資料集建立並進行相關處理與測試。

並進行討論後，設立進一步的實作目標與分工，完成實作與統整。

面相有像是了解不同 features 對一項指標（即稻米產量）的影響或是重要程度，或像是掌握各地區的緯度、氣溫、日照時數、雨量，進而判斷該地的稻米產量相關問題，如不同狀況下可能會有的產量，或是明年可能會有的產量等等，可以往資料預測發展。

2.資料來源：

(1)台灣稻米資料：農業資料開放平台

(2)台灣水文資料：水利署中文版全球資訊網

(3)台灣氣候資料：CODiS 氣候資料服務系統、TCCIP

二、個人報告簡要介紹

1.李家維：

(1)實作目的：

藉由資料建構 Ensemble Regressor，透過給定各項 feature 數值時，預測來年雲林縣的稻米產量。

(2)Data mining 模組、改變控制參數說明：

這裡目標是透過使用 scikit-learn 中的 MLPRegressor()和 RandomForestRegressor()進行產量稻米的回歸預測。

```
from sklearn.neural_network import MLPRegressor
from sklearn.ensemble import RandomForestRegressor
```

(3)程式、環境設定,執行方式說明：

程式語言使用 Python，環境使用 Visual Studio Code 搭配 Python 3.8.8。

```
# 選擇特徵
features = [
    'pressure(hPa)',
    'sea_level_pressure(hPa)',
    'min_pressure(hPa)',
    'max_pressure(hPa)',
    'temperature(℃)',
    'max_temperture(℃)',
    'min_temperture(℃)',
    'wind_speed(m/s)',
    'max_wind_speed(m/s)',
    'precipitation(mm)',
    'max_precipitation(mm)',
    'ground_temperture_0cm(℃)'
    ]

# 分割特徵和目標變數
X = merged_data[features]
y = merged_data['產量']
```

(4)評估方法：

這裡使用標準化後的 MSE（Mean Square Error）評估。

$$MSE = (1/n)\sum_{i=0}^{n} (y_{pred} - y_{true})^2$$

(5)結果與討論：

由於最後只有 18 筆(2005~2022)資料做訓練，目前只好採用隨機生成 10 筆資料做測試，但也可以產出合理範圍內的資料。

使用訓練集測試時，MLPRegressor 和 Random Forest Regressor 則分別有 147.79 和 0.378 的標準化 MSE 分數。標準化 MSE 分數在 0~1 之間代表有良好貼近資料，大於 1 則是有較大誤差。Random Forest Regressor 為 Decision Tree 的集合體，較易上手操作。MLPRegressor 則有較多可以調整的元參數可做調整，但也需要較多的經驗來做修改。

當中，在測試隨機生成資料時，MLPRegressor 和 Random Forest Regressor 分別有 25.18 和 218.90 的標準化 MSE 分數，推測 Random Forest Regressor 較貼近訓練數據，也就離隨機生成數較遠。相對的，MLPRegressor 也表現出較有靈活應對的感覺。

2.陳奕廷：

(1)實作目的：

藉由資料建構 ensemble classifier，判斷在給定各項 feature 數值時，各種稻米的單位產量是否高於一定的數值，如粳糯稻(圓糯)的單位產量是否大於 5。

(2)Data mining 模組、改變控制參數說明：

首先是有進行以 Missing data 將空值補 0 與以 Binarize 設立相關閥值，建立新的 class（class 中的資料為二元化內容），還有將資料為 0 的部分 drop 掉不進行資料分等三項資料前處理。

Data mining 模組則是有使用：

1.RandomForestClassifier() →Accuracy=0.33

2.LogisticRegression()→Accuracy=0.33

3.DecisionTreeRegressor()→x

4.DecisionTreeClassifier()→Accuracy=0.67

5.SVC()→Accuracy=0.33

6.MLPClassifier()→Accuracy=0.67

7.GradientBoostingClassifier()→Accuracy=0.67

8.KNeighborsClassifier()→Accuracy=0.67

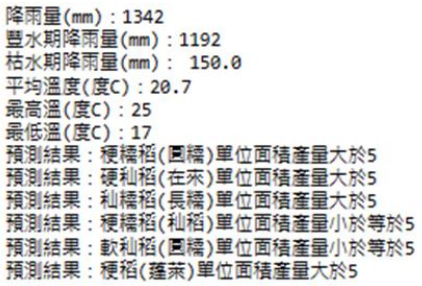
但其中由於 DecisionTreeRegressor()的 R^2 score 為 0，對於這個問題解釋能力有限，加上 DecisionTreeRegressor()並不適用二元分類的準確度評估指標，對於二元分類的解釋度較低，因此我的 voting classifier 就將其拔除了，相應的我則是使用能做 accuracy 判斷，且可以、適合用於二元分類的剩下五個 classifier 進行探討，而從 accuracy 與 Classification Report 的分析中，我也將效果較差的 classifier 移除，將剩餘的 classifier 進行 voting classifier 的建構（最後有使用的 classifiers 有 DecisionTreeClassifier()、MLPClassifier()、GradientBoostingClassifier()、KNeighborsClassifier()）。

最後的 voting classifier→Accuracy=0.75

(3)程式、環境設定,執行方式說明：

以上方的五個 classifiers 去建立 ensemble classifier 後，便能以這個 voting classifier 去進行 class 的判斷，而判斷前我們則是能輸入各項 features 的數值，即降雨量、豐水期降雨量、平均溫度、最高溫、最低溫，接著便能進行預測。

如圖所示：



(4)評估方法：

我的評估方法主要都是以 accuracy 去做判斷，了解各個 model 以及最終 model 的 accuracy 為何，同時也以各個 model 的效果去選擇最後要使用 classifiers。

(5)結果與討論：

首先就是從結果而言，ensemble classifier 的 accuracy 較高，對比只使用單一 classifier 進行預測，結果會更好。

而在程式中，我們可以選擇有相同 features 與 targets 的資料集進行模型建立，不是只能使用我的這份資料集，同時閥值的選擇也可以建構出不同模型，進行預測的範圍其實蠻廣的，不過這些內容若想要做介面處理進行輸入則是未來有機會可以製作的，本次並沒有執行到那部分。

同時也能藉由 accuracy 的判斷選擇要用的 classifier，未來也可以做成彈性調整的 ensemble classifier，如 ensemble classifier 能有效地進行多個 model 的建立，以符合不同 target 的需求，也能實際的進行預測，當有更多筆資料進行訓練後，應該可以有更好的 accuracy 與表現。

3.李承恩：

(1)實作目的:

建構 regressor model, 預測台南稻米產量，了解哪些因素會影響稻米產量。

(2)Data mining 模組、改變控制參數說明：

我用的模型有: SVR, Random Forest Regressor, Gradient Boost.

在用 SVR model 的時候有改變使用的 kernel. Random Frest Regressor 和 Gradient Boost 則是觀察在不同 n_estimator 的情況下的模型分數。

(3)程式、環境設定,執行方式說明：

pre-processing 的部分:

由於氣候資料是以月為單位，我先將他們合併成以年為單位。

使用的氣候資料集剛開始只有包含 **年雨量， 年度平均氣溫，年度平均大氣壓，平均溫度。**但後續在做 feature selection 的時候發現他們與產量的 correlation 不高，只有 0.1~0.2 左右。 後來我使用了更完整的資料集（約 20 個 feature），包含日照、濕度、紫外線等等。又進行了資料型別的轉換與 feature creation, 如整年的總日照量 (MJ/㎡)，總降雨時數，總降雨日數等等。

	年度	年度降水量	年度降水日數	年度降水時數	平均氣溫	平均海平面氣壓	平均相對溼度	年度日照時數	年度天空日射量	年度平均雲量	平均日最高紫外線指數	平均當月最高紫外線指數
count	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000
mean	2013.000000	1924.464706	85.411765	370.447059	24.774020	1012.913725	74.578431	2225.711765	5802.977647	5.296569	6.906863	9.387255
std	5.049752	565.068001	14.226115	101.657815	0.324704	0.449671	2.173361	196.435240	587.909955	0.285820	0.772735	0.910939
min	2005.000000	1195.200000	57.000000	237.300000	23.991667	1011.908333	71.166667	1973.200000	4600.470000	4.816667	5.916667	8.083333
25%	2009.000000	1481.000000	73.000000	276.400000	24.566667	1012.658333	73.083333	2071.600000	5380.330000	5.141667	6.333333	8.666667
50%	2013.000000	1867.200000	86.000000	373.500000	24.833333	1012.916667	74.416667	2171.900000	5989.940000	5.266667	6.750000	9.083333
75%	2017.000000	2241.500000	97.000000	424.500000	24.983333	1013.258333	75.583333	2352.600000	6237.310000	5.533333	7.416667	10.000000
max	2021.000000	3148.500000	107.000000	566.700000	25.275000	1013.575000	80.416667	2595.900000	6520.100000	5.775000	8.500000	11.083333

在加入更完整的資料集並做 feature creation 之後，可以觀察出這些 feature 與產量的 correlation 更高了。如**年度降水時數**的重要性就遠高於**年度降水量**與**年度降水日數。 相對濕度**(0.398) 對產量的影響也是當初我沒想到的。

```
Index(['年度降水時數', '平均相對溼度', '年度天空日射量', '平均當月最高紫外線指數'], dtype='object')

年度降水量            0.167778
年度降水日數            0.088614
年度降水時數            0.388655
平均氣溫              0.145669
平均海平面氣壓            0.058332
平均相對溼度            0.398063
年度日照時數            0.048402
年度天空日射量            0.442835
年度平均雲量            0.201082
平均日最高紫外線指數        0.200354
平均當月最高紫外線指數       0.244006
dtype: float64
```

我在不斷的微調 threshold 後 篩選出四個 feature 來訓練模型，得到的模型結果最好 (年度降水時數, 平均相對溼度, 年度天空日射量, 平均當月最高紫外線指數)。而產量與氣候相關資料的數量級相差有點大，所以我又對 dataset 做了 Normalize, 用的方法是 Min-Max Scaling.

模型訓練的部分:

用了 SVR, Gradient Boost Regressor, Random Forest Regressor.

有無進行 feature selection 訓練模型的 MSE 數值如下:

無　feature selection -> 有 feature selection

SVR: 0.17 -> 0.09

Gradient Boost: 0.2383 -> 0.0876

Random Forest: 0.2832 -> 0.0732

可以發現進行 feature selection 的確使模型的表現更好了。

SVR 的部分我嘗試使用不同的 kernel, 發現 poly 的效果最好, MSE 約 0.09, 其他如 linear, sigmoid, rbf 等等 MSE 都落在差不多 0.17. Train test split 的部分我保留 30% 做 testing, 發現訓練出的模型表現最好。

(4)評估方法：

對 feature 和 target(產量)做相關分析，選取前四高分的特徵訓練模型。模型評估用的是 Mean Square Error. 我後來嘗試用訓練出的模型對其他年份(非原本資料集的部分)和其他縣市做預測，用的是 R^2.

(5)結果與討論：

在最後評估模型的時候我嘗試用訓練出的模型對其他縣市/年分的資料做預測， MSE 一樣表現不錯 (約 0.06) 但 R^2 為 -1.6. 後來詳細觀察發現可能原因是資料集太少，而且資料完整性不足。由於許多縣市的觀測站是近年才建立的且很多測量的值有缺失。我負責的區域(台南)是資料較完整的區域，推測是這樣導致實際測量的結果沒那麼準確。

三、發現與整合

1.預測方式：

我們每個人使用的模型並不一樣，有人是使用 regressor，有人是使用 classification，其實就是因為想要呈現與解決的問題並不相同，但這也增加了我們整合的多樣性，我們可以解決不同的問題，以報告為例，我們的預測就能藉由不同處理器預測實際的稻米產量，也能使用 voting classifer 去預測分類，瞭解單位面積稻米產量是否超過一定的閥值。

另外就是在預測稻米產量時，我們也能用不同的 regressor 去做預測實際產量，加強最終預測結果的可信度，也能計算單位面積產量後，訂定閥值，並由 voting classifer 預測的結果再一次的驗證預測是否準確。

2.可以用於不同資料集：

由於我們的整個方案撰寫都是從資料集的前處理、模型建構到最後的結果預測，因此我們也能將有相同 features 與 targets 的資料進行整理，變成資料集後，放入程式進行個別的資料集前處理、模型建立與結果預測，能應對不同資料集有各自的呈現，讓使用的範圍更廣。

3.不同評估方式：

同時我們使用的評估方式也都有所不同，解決回歸問題使用的是 r^2 score 及 mse，分類問題則是使用 accuracy、Classification Report，但使用的評估方式都不只有一種，其實也能讓模型的建構更為完善，也能藉由分析去理解不同模型的優缺點，並做出模型的選擇與建構，增加結果的可信度。

4.不同 kernel 的影響：

SVR model 中 有 linear, sigmoid, rbf, polynomial. 在對進行 feature selection 後的資料進行 train/test, 發現 poly 的效果最好. MSE 約 0.09, 其他 kernel 訓練出來的模型 MSE 落在約 0.17。

5.feature selection：

我們雖然使用的資料來源都是相同，但在進行資料處理成要用的資料集時，會發現有些測站並沒有相應資料，因此使用的 feature 就會有所不同，而我們發現若有較多的 feature，我們也能進行 feature selection，能讓整體預測效果更佳。

同時 feature selection 也是讓我們能夠進行多一次的驗證，我們能藉由相同的模型，去處理 test data，看看結果的成效是否比沒有進行 feature selection 或是不同 feature 的選擇為佳，更加確定 feature 與 target 之間的關聯性。

四、結論

1.優點與缺點：

整合後，其優點為我們整體能有較為多元的預測方式、評估方式，同時資料集的使用只要符合相對的 features 與 target，就能丟入程式進行前處理、模型建立與結果預測。

另外我們也發現很多部分有做不同的選擇或處理，會有不同的影響，這些內容在進行處理與分析後，都能讓模型在預測時有更好的發揮，未來進行加強時也能圍繞這幾個部份去做加強。

缺點的部分則是我們的輸入介面其實沒有到很完整，因此有些內容的選擇要從程式去做修改，並進行手動調整，而不能從輸入介面去進行選擇。

另外針對一些評估方式的分數，其實沒有到非常好看，或許也可以想辦法去做加強，得出更好的評估分數。

2.未來進步方向：

首先是資料集選擇部分，由於選題問題，我們的資料集選擇範圍有限，加上我們是自己蒐集網頁資料進行資料集的建立，因此資料量跟資料完整度相對不足，未來可以想辦法去建構資料更多且更完善的資料集，訓練與測試的成果也能更好。

接著是可以對更多不同的 regressor 跟 classifier 進行分析，並用不同評估方式去了解不同模型對於問題解決的優劣程度，並進行進一步的分析、合併與整合；同時 ensemble classifier 的建構方式也可以更加多元與嘗試，像是我們有組員是以 voting classifier 中 voting='hard'作為 ensemble classifier，那或許也可以將不同的 classifier 進行重要程度的判斷，變成不同的投票可以有不同權重等等，或是也能將 regressor 嘗試進行 ensemble，並且進行有權重或無權重的預測。

然後是輸出輸入介面呈現，我們可以針對實測內容去設計更良好的輸出輸入介面，同時輸入的資料也可以更多元，讓整體的模型建構更有效能或是符合使用者需求，一些部分的選擇也可以更加多元（如讓使用者自行選擇要使用的

模型，或是模型的 accuracy 要到多少以上再使用等等），變成使用者與整個模型有更多的互動。