Data Mining Project 2 Report

學號：P76114171

姓名：羅子翔

- 問題描述
  - 如何製作一個好的厚皮 pizza？

    利用上課所學，設計 5 個實驗，每個實驗都使用 Decision tree、SVM、KNN、Naive Bayes 分類，並且顯示 tree 的結構以及用 PCA 降維的方式將資料分布可視化，好壞 pizza 的比例是 1：1。
- 資料設計
  - 什麼是一個好 pizza？

表 1 absolutely right rules

屬性名稱	使用量
高筋麵粉	140±10g
低筋麵粉	100±10g
溫水	140±10g
糖	6±1g
酵母粉	5±1g
橄欖油	15±10g
鹽	3±1g
靜置時間	30±10min
溫度	220±10°C
烘烤時間	9~13min

表 2 variant

屬性名稱	機率
Extra effort (ex.番茄醬、額外調味、裝飾、etc.)	90%的好 pizza 會有這個屬性
鳳梨	10%的好 pizza 會有這個屬性
珍珠	10%的好 pizza 會有這個屬性

表 3 non-relevant

屬性名稱	數值
Elec power	電力穩定度，0 或 1
Cooker age	廚師年紀，18~100
Cooker gender	廚師性別，F 或 M

- 什麼是一個壞 pizza？

  absolutely right rules：每個百位數的屬性值上下界±100；十、個位數的屬性值±10，若是壞 pizza 的屬性值落於好 pizza 的區間則會再重新 random 數值，直到不在該區間內。

  variant：Extra effort 機率 90%→10%；鳳梨、珍珠機率 10%→90%。

  non-relevant：相同。

- 資料前處理

1. 將廚師性別的 F、M 轉成 0、1 數值。
2. 每個好 pizza 有 10%機率會變種成壞 pizza(雜訊)。
3. 每個壞 pizza 有 10%機率會變種成好 pizza(雜訊)。

- 實驗結果與說明

- 實驗一，大 train、test (6666、3333 筆)+小雜訊 (1%機率 mutate)

  在實驗一中，可以看到雖然雜訊只有佔資料的 1%，以 PCA 視覺化可看到分布還算集中，但是 decision tree 產生的規則就已經很雜亂了，雖然在訓練資料集 predict 的部分還算是正確，但我想若是在資料分布更複雜、類別數更多(在此例中只有 2)，而屬性數量不變的情況下，我想在相同訓練資料筆數與雜訊佔比的情況下，這棵 decision tree 應該是會 overfitting 的，可見 decision tree 的形狀(分支)與雜訊高度相關，顯示其的不穩定性。

  而其他的方法(SVM、KNN、Naive Bayes)，也都還有不錯的結果，因為這時資料分布較集中，所以 KNN 和 SVM 在高維空間中表現較不錯，而 Naive Bayes 也有充足的資料量做預測。

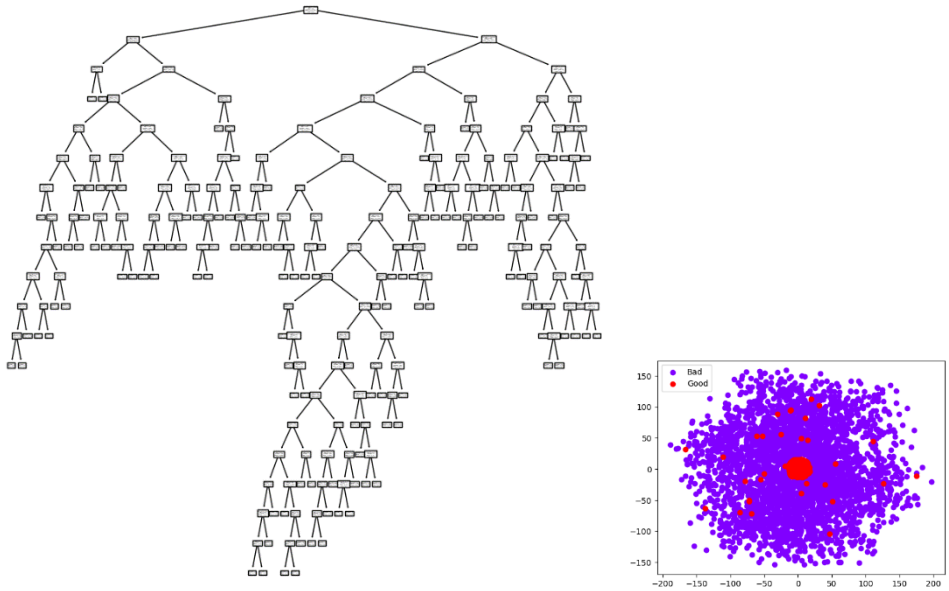

圖 1 E1 decision tree & PCA

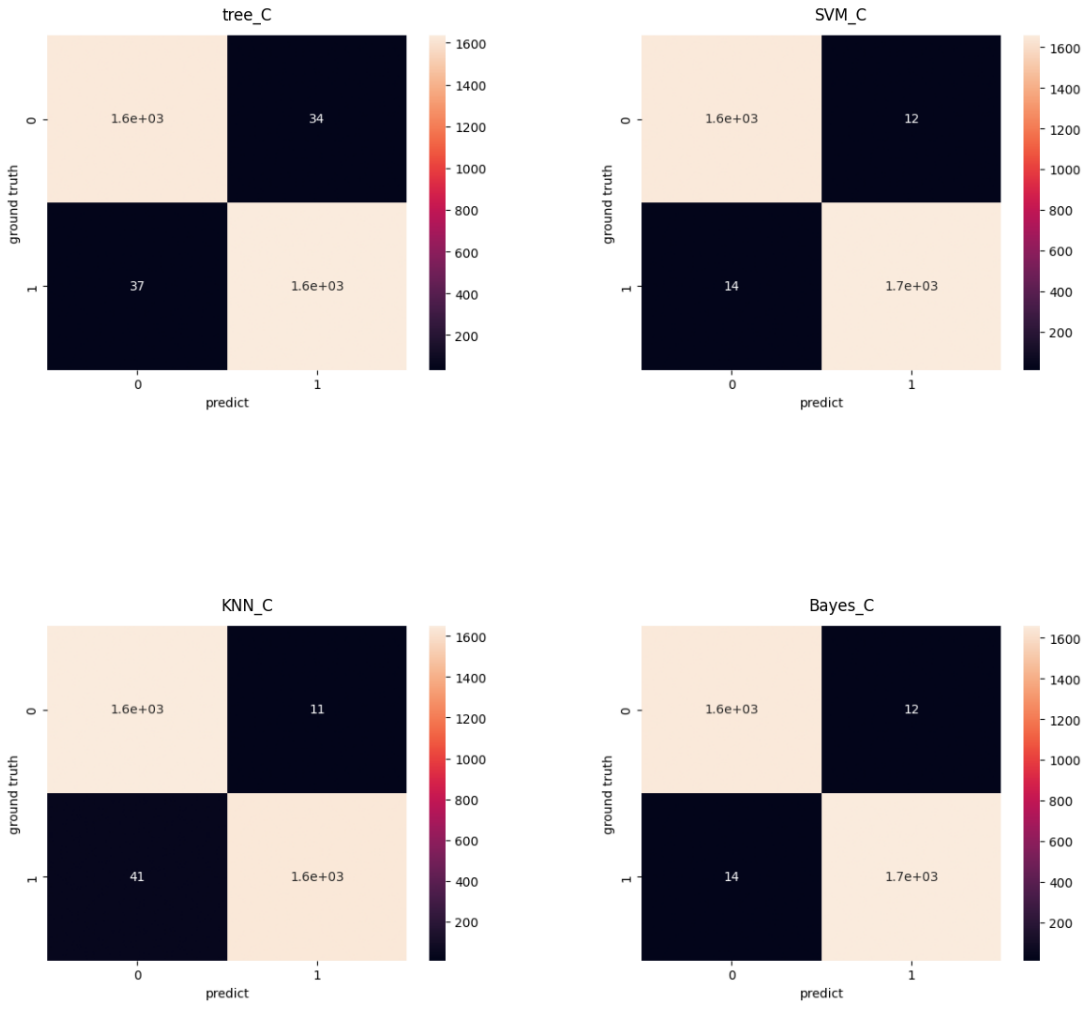

圖 2 E1 confusion matrixes

- 實驗二，大 train、test (6666、3333 筆)+大雜訊 (10%機率 mutate)

在實驗二中，訓練與測試的資料數量不變，但是增大了雜訊佔比，可以看到，決策數的長相極度不堪入目，且 FN 與 FP 的數量急遽上升，這是一個很明顯的 overfitting，在 PCA 上也呈現了資料較分散的趨勢，不過 10%的雜訊已經對 decision tree 產生很大的影響了。

而在 SVM、KNN 和 Naive Bayes 的方法中，FN 的數量變得很大，但 FP 卻還是維持一個較小的數值，accuracy 降到了約 8-9 成，而 recall 值(TP/(TP+FN))降到了約 7 成，代表有 30%的 positive data 沒有找到，且因為這個資料分布 positive data 較為分散，因此 KNN 在這個情況，FN 的數量比別人多了一些。

若是再進一步的把雜訊調成 30%，可以發現大家都變很糟，但相對於其他演算法 FP 和 FN 數量幾乎相等，KNN 有讓錯誤集中在 FP 的趨勢，我想這是因為 positive data 還是相對集中，但是中心也同時出現了很多的雜點導致將 negative data 誤判為 positive，導致了 FP 的上升。

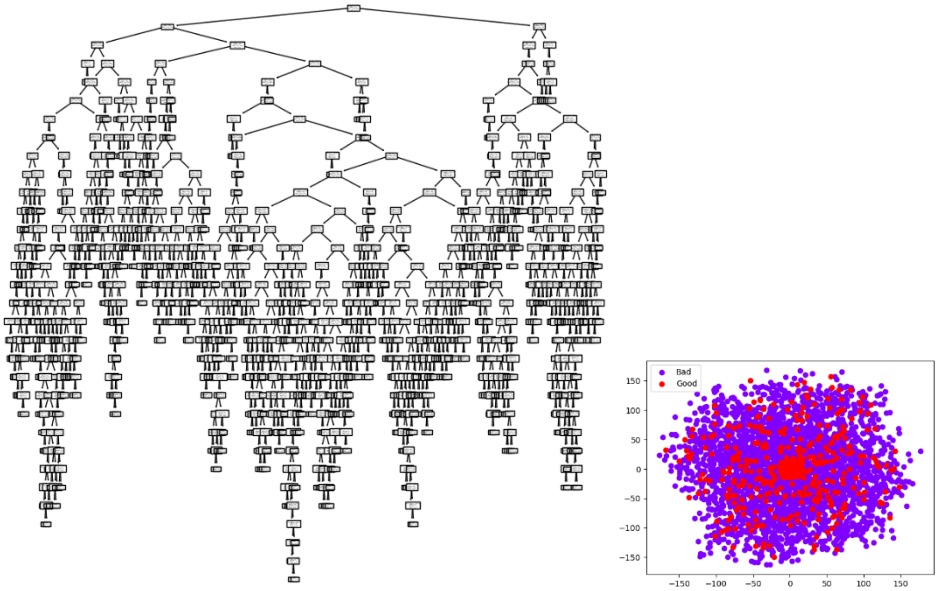

圖 3 E2 decision tree & PCA

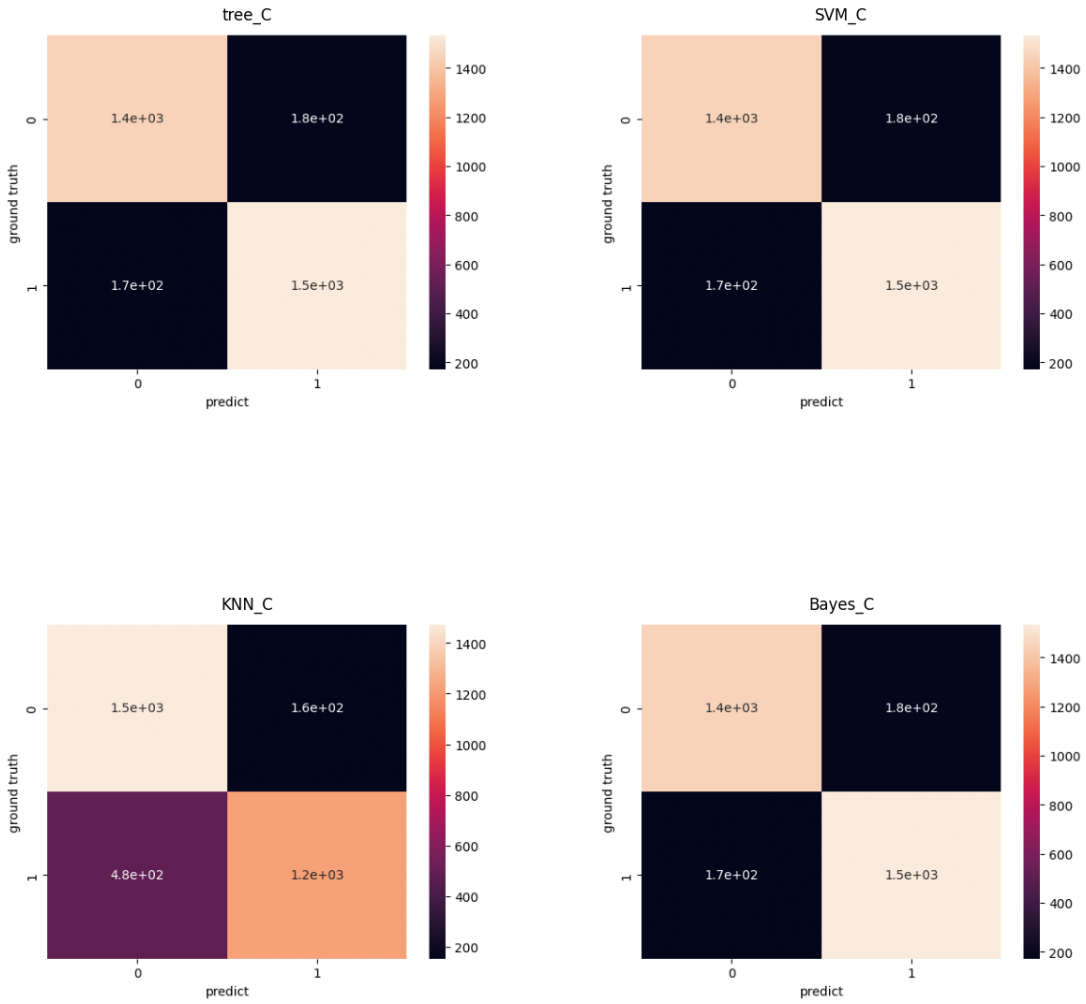

圖 4 E2 confusion matrixes

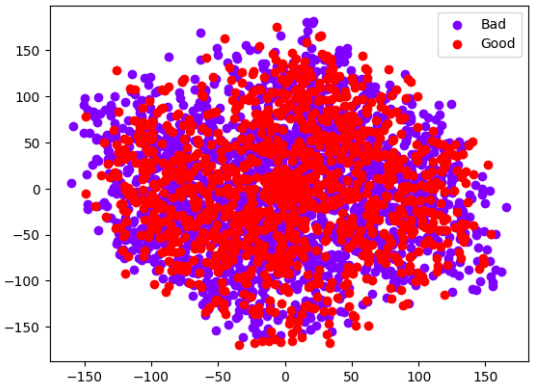

圖 5 雜訊 30%的 PCA 分布

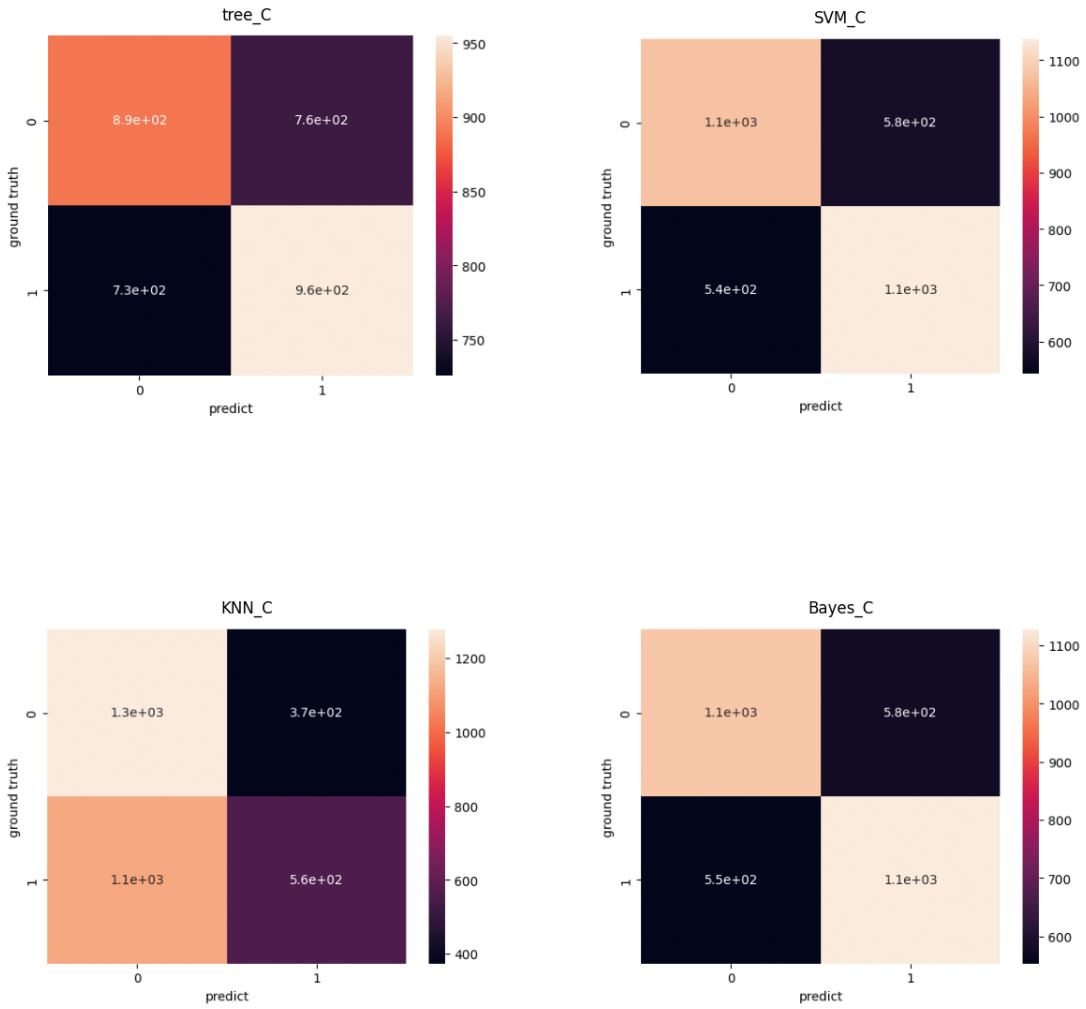

圖 6 雜訊 30%的 confusion matrixes

- 實驗三，小 train、test (666、333 筆)+小雜訊 (1%機率 mutate)

  在實驗三的小資料小雜訊的情況中，decision tree 總算是稍微輕鬆一點的找到 pattern 了，PCA 也呈現出了一個很簡單的分布，其他的方式正確性也都很高(因為雜訊依照機率只會出現個位數筆)。

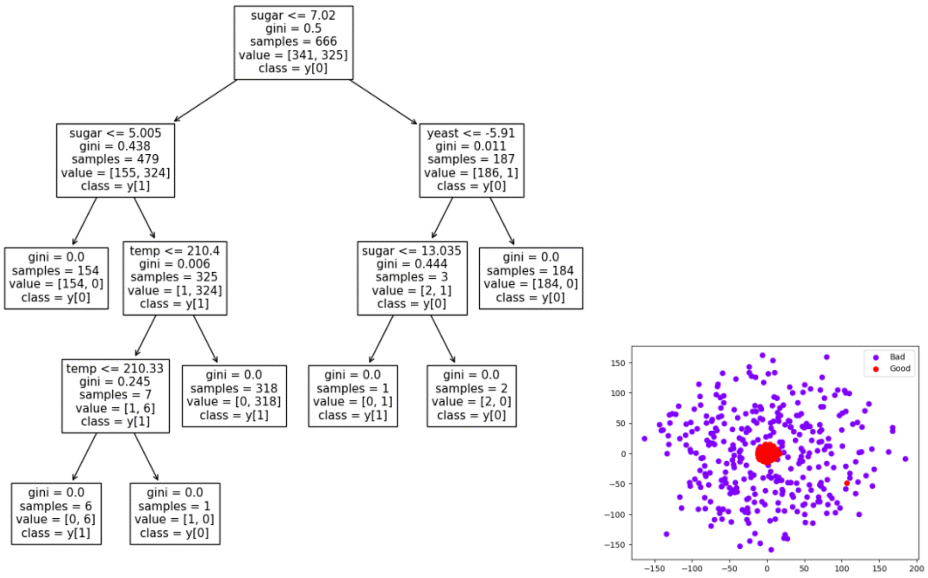

圖 7 E3 decision tree & PCA

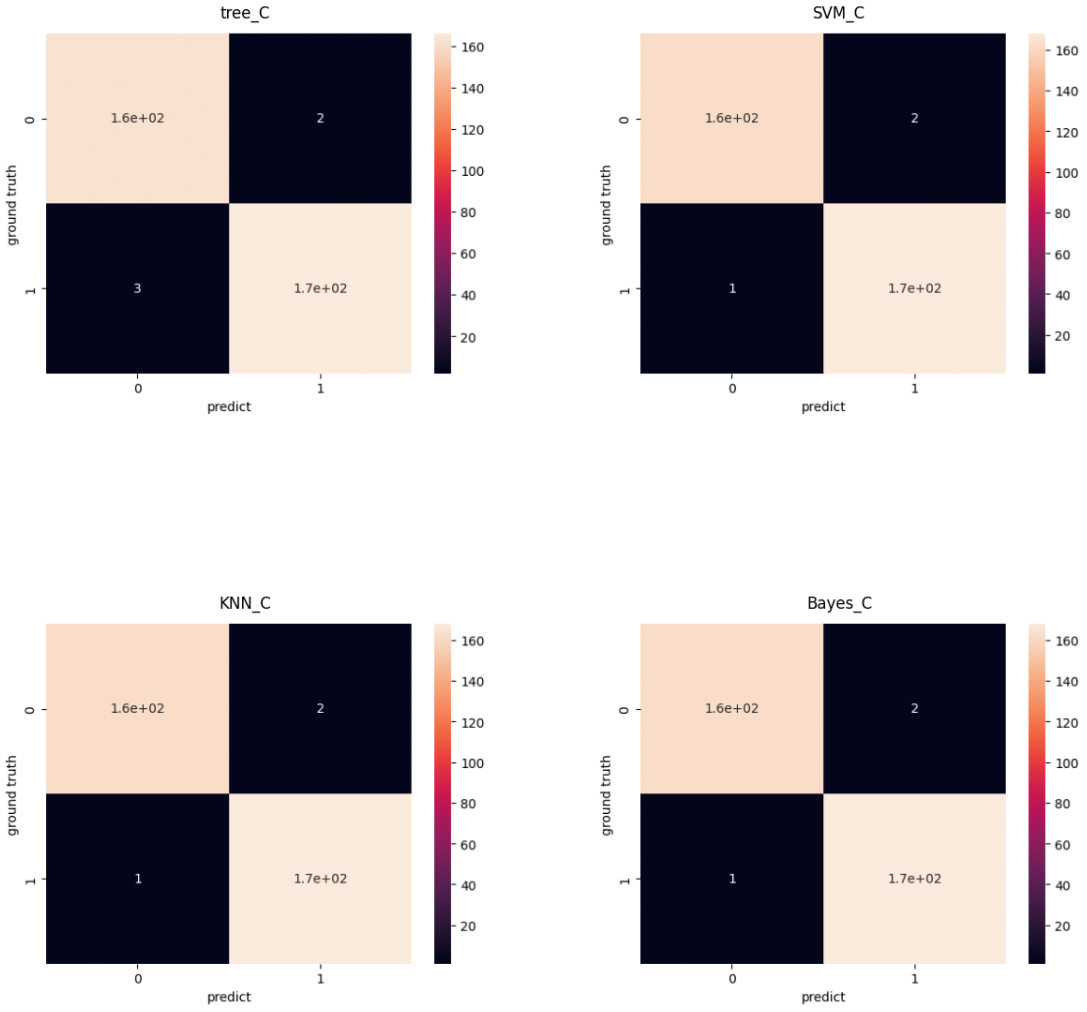

圖 8 E3 confusion matrixes

➢ 實驗四，小 train、test (666、333 筆)+大雜訊 (10%機率 mutate)

整體來說在實驗四中，每種演算法都表現得還不錯，但是 decision tree 又明顯的變大棵了而且也表現不太好，且 KNN 在這個情況也同實驗二的描述，FN 的數量比別人多了一些。

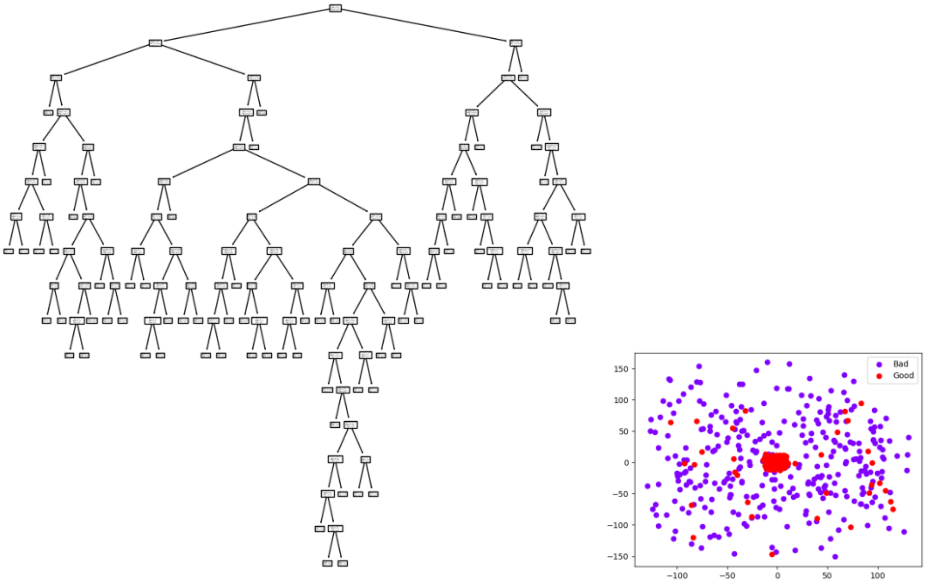

圖 9 E4 decision tree & PCA

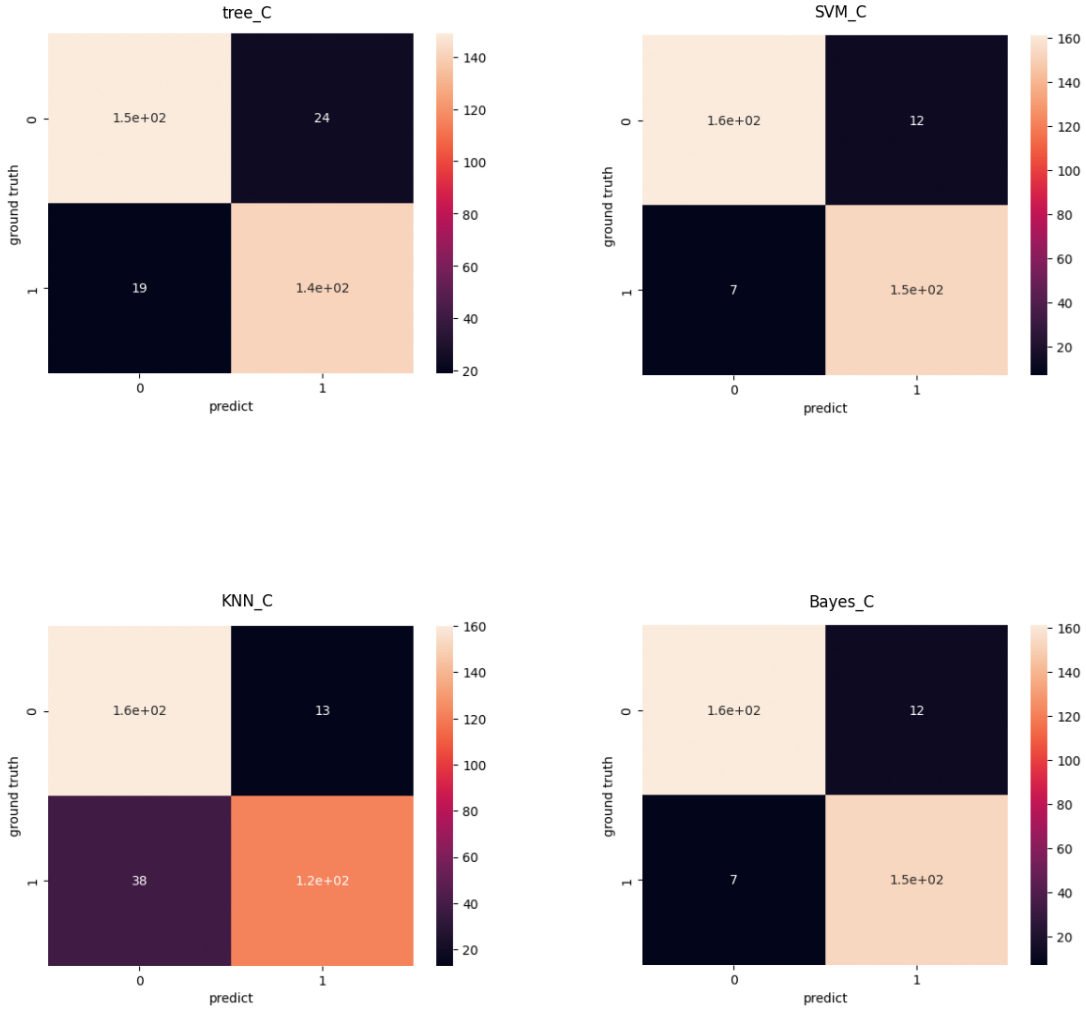

圖 10 E4 confusion matrixes

- 實驗五，更改資料分布，與實驗一~四比較

實驗五我希望能加強類別間的差異，讓 decision tree 在大資料中快速找到一個好的 pattern。結果發現，不管我如何將資料的特徵數值區分得更明顯，甚至都快將答案偷偷藏在訓練資料中了，但是雜訊就算只有 1%，decision tree 還是會長的很大顆，直到我把雜訊調到 0.03% 才有一個比較好看的 tree，雖然這也有可能是我的資料設計的不夠嚴謹，不過相對來說，其他的方法都比 decision tree 穩定，也比實驗一到四好，在這個情況中都表現得不錯。

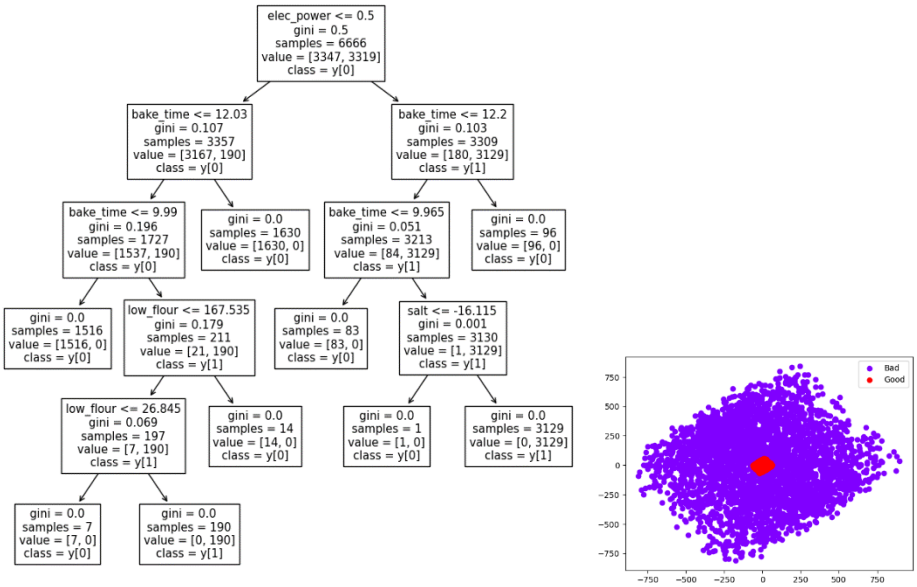

圖 11 E5 decision tree & PCA

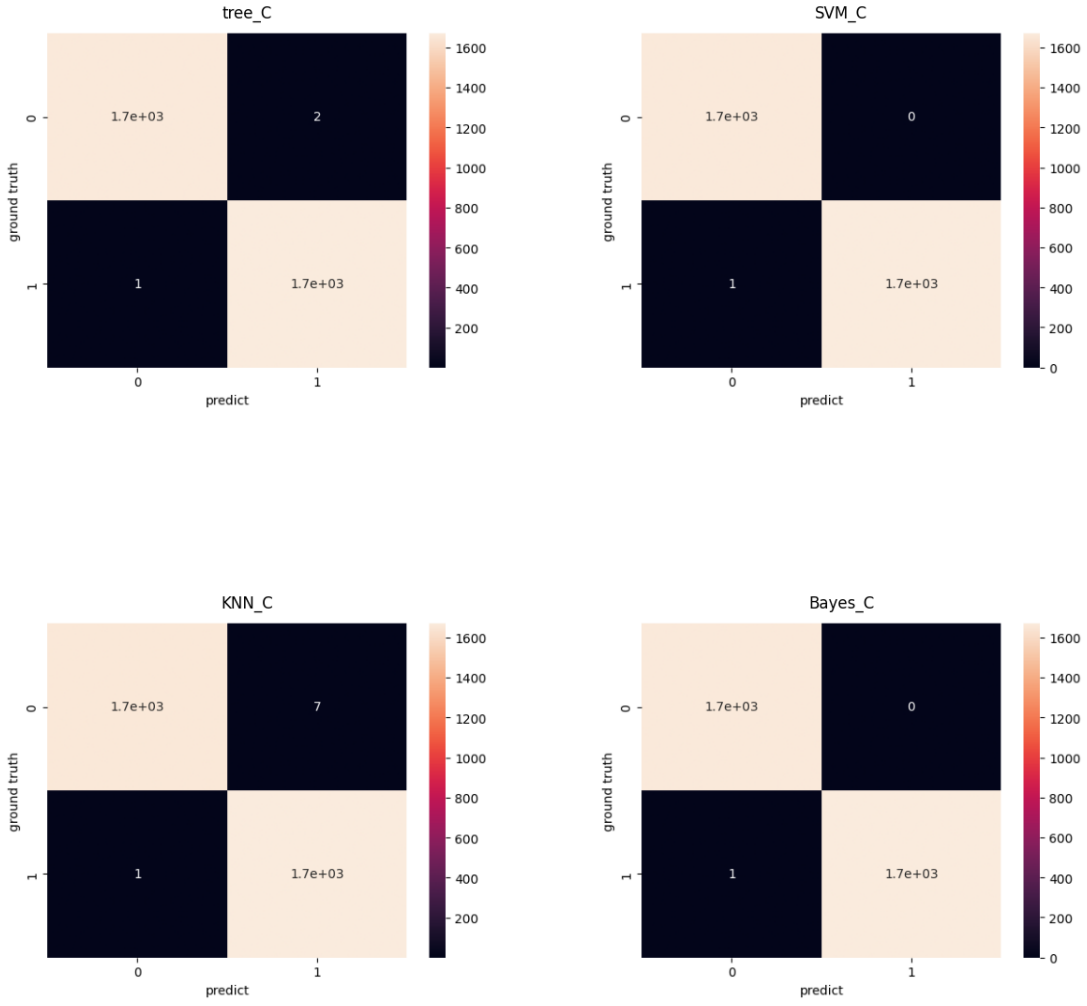

圖 12 E5 confusion matrixes

- 結論

  經過實驗一~五，經過了多次修改資料屬性值的分布與雜訊的比例，發現 decision tree 的結構的確受到雜訊很大的影響，有時候會找出一些莫名其妙的規則，感覺一個不小心就會 overfitting 了，所以後來才有 random forest 的做法，讓結果比較不會 overfitting。

- 可改進的地方
  - 資料分布可改為非球型分布。
  - 雜訊分布可改為高斯分布。
  - 可以增加 outlier

  不過這三點比較難在高維度的情況下人為生成這些資料分布，所以這次作業(要求 10 個 attribute)應是無法實作。