Data Mining Project 2

Classification

資訊所碩一 P76074575 潘昌義

一、 作業簡述

二、 資料集製作與評估

三、 Decision tree

四、 Random Forest

五、 討論與延伸

六、 總結

一、 作業簡述

在主題三中，我們為了能區別不同種類 item 的相異性，使用不同 classification 的方式嘗試將 data 正確分類。在這個作業中，我們需要自己製作或找一個資料集，且這個資料集具有一個絕對的分類方式(absolutely right rule)，以方便我們未來檢驗這棵樹的正確性。關於演算法的部分，我們可以使用 sklearn 裡面的模型，例如 decision tree classifier 或 random forest classifier 等方法，最後使用 graphviz 將結果繪製成圖，並檢驗是否與自己原先假設或資料給定的 absolutely right rule 相符或類似。

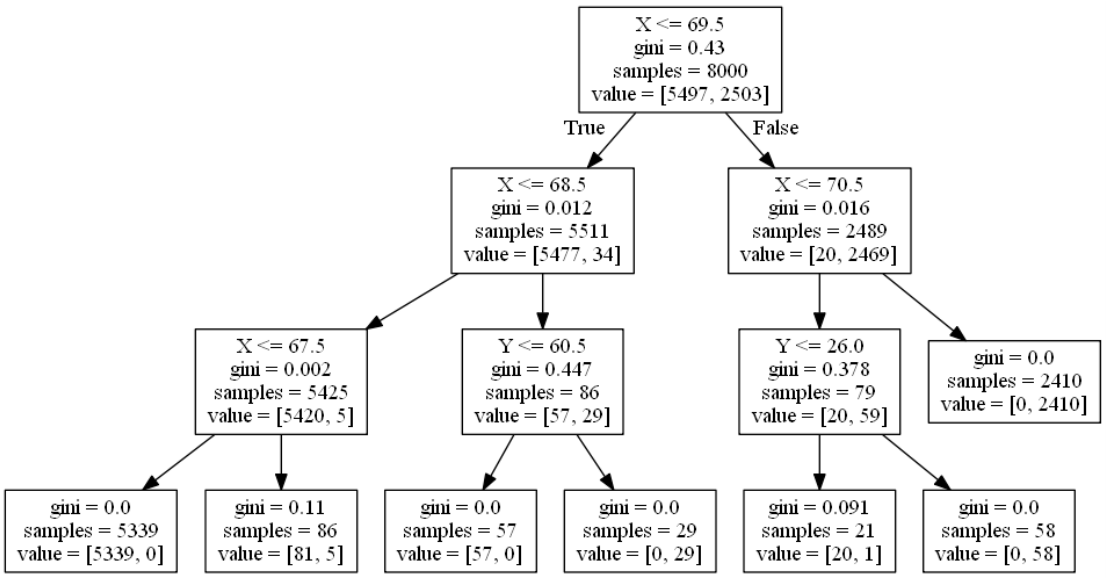
二、 資料集製作與評估

由於我們需要一個具有 absolutely right rule 的資料，因此我選擇自己以一個簡單的數學方程式來進行資料的生成，其中加入一點點的小誤差使資料不會完全按照 absolutely right rule 走，以觀察稍晚的模型能不能不被這點”現實因素”所影響。

為了使最後觀察結果簡化，我選擇採用兩個特徵 X 與 Y，構成一個二元二次方程式$x^2 + 4y + b$，對每一個變數 X 和 Y 都取一個介於 1~100 的隨機值，其中 b 是一個隨機震盪的數值，介在-3~3 之間。由於我們可以由方程式變數的定義域，先行推斷值域位於-1~10403 之間，因此我假設分類的觀察結果為”該數是否大於 5000”，若大於 5000 則設為 1，反之設為 0。

三、 Decision tree

我們以 sklearn.tree 裡面的 DecisionTreeClassifier 將建好的資料進行分類，分類完成後以同樣在 sklearn.tree 裡面的 export_graphviz 將結果輸出成.dot 檔，最後再以 graphviz 將圖片輸出，其結果如下：



我們不難從上面看出，decision tree 的第一層分類馬上幫我們從 X=69.5 的地方區分為左右兩側，對這個結果不會感到太意外是因為我們對 X 做了平方的處理，自然他的比重會相對較大，另外$\sqrt{5000}$ = 70.7，也就非常接近上述分類的 69.5 了。

至於為什麼會比 70.7 來得小呢？其實原因很簡單，因為我們在 Y 的部分配了一個 1~100 的隨機數，對方程式的整體獎勵是 4~400，因此要達到 5000 這個值會比原先還要來的簡單一些，所以 X 就比 70.7 來得小了。

四、 Random Forest

除了上面用到的 decision tree 外，在課堂上我們也有提到隨機森林(Random forest)是個良好的分類器，他是由多個 decision tree 所構成的分類模型，藉由設定 tree 的數量來提高精確率，但需要付出的代價就是運算所需的時間。我們從 sklearn.ensemble 內將 RandomForestClassifier 引進程式後，除了將樹的數量設定為 10 之外，其餘設定不變跑相同的 training data，以相同的 testing data 測試不同分類器的 score，結果如下所示：

```
In [9]: tree.score(X_test, Y_test)
Out[9]: 0.998

In [10]: forest.score(X_test, Y_test)
Out[10]: 1.0
```

雖然沒什麼進展，但還是比較準哪

由於 random forest 是由多個 decision tree 所構成的，所以如果要做視覺化的圖就需要個別對每個 tree 做圖，~~太麻煩了可以不要做嗎。~~

五、 討論與延伸

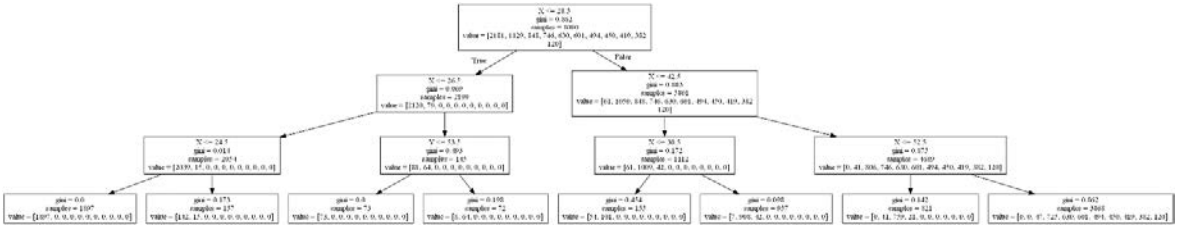
除了完成上述的測試外，這邊分享一下我前面嘗試的奇怪(？)組合：

1. 分類”餘數”

不得不說這個真的不太行，因為餘數不能由數字的大小來直接判斷，所以分類器做出來的效果之差無法想像，用 decision tree 分類的 socre 只有 0.145 分，而他的圖呢...~~直接砍掉重練了...~~

2. 分類”商”

前面都試過餘數了，當然會想試商數阿，然而這個分類結果卻是非常的壯觀，不是說他不準，而是圖的規模莫名其妙的大...

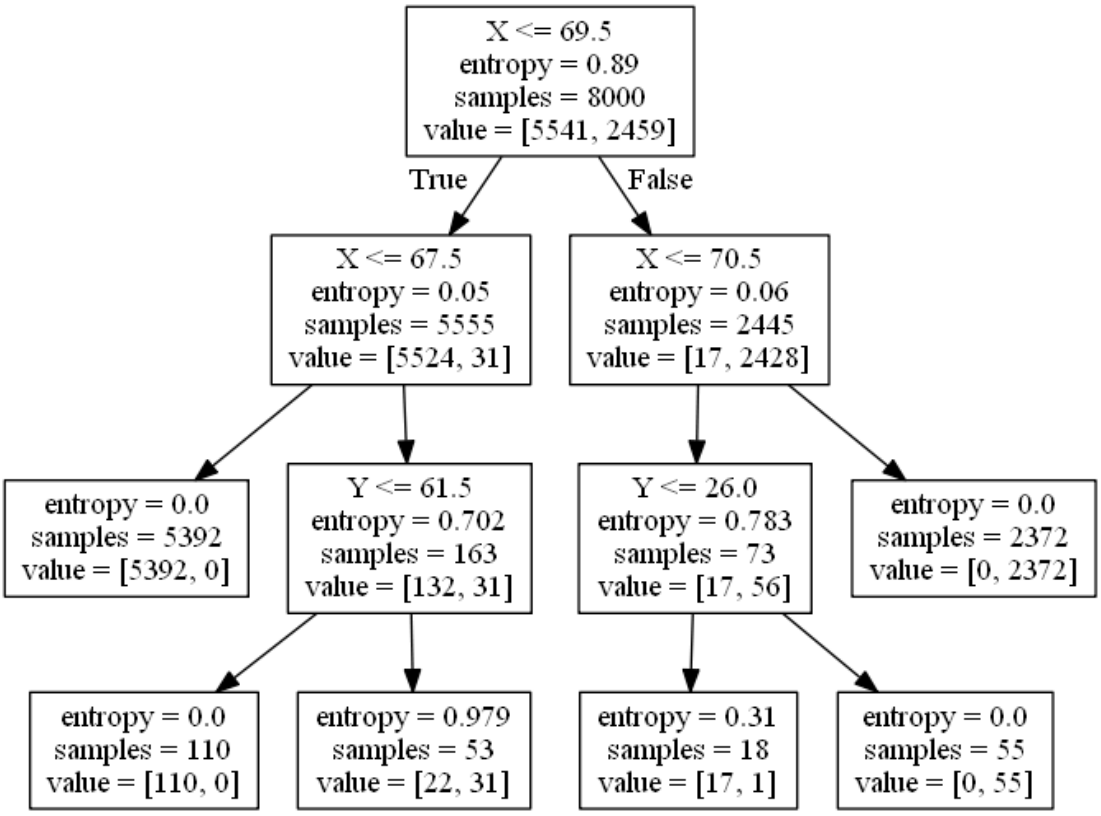


有附檔，~~有興趣可以點開看看~~

我想會出現這種問題大概是因為我的數值介在 0~10000 之間，而我又把 1000 作為一個單位，自然出現的結果就大到爆了...

3. 嘗試以不同的 criterion(分類標準)做

模型的預設基本上都是 gini，這邊我選擇改用 entropy 作 criterion，其結果如下：



其實還是能夠清楚的看出在第一層與第二層能夠輕鬆以 X 的值區分類別，而底下用 Y 區分導致 entropy 較高也在我們的預期下，~~畢竟本來我就沒打算要模型用 Y 去辨識了。~~

六、 總結

整體來說，我們可以看到 absolutely right rule 中我們假定影響比較大的 feature (例如我擬定的 X，他是平方項)，對分類結果的影響較大，在不同的分類標準下皆在第一層就優先被區分了；而比較不重要的 feature (例如僅有一次項的 Y)則會被放到底層的部份去做進一步的區分，然而它的效果通常不太好，可能是因為深度不夠，也有可能是因為本來就很難拿一個冪次較低的特徵來做良好的分辨。

此外，上述實驗有個小小的失敗，就是我們幾乎沒能看出誤差項 b 對整體分類是否有構成影響，我想大概是因為我設的數字太小了吧，反而 Y 比較像是這個實驗的誤差項，觀察一次項能否影響二次項對整體結果的改變。