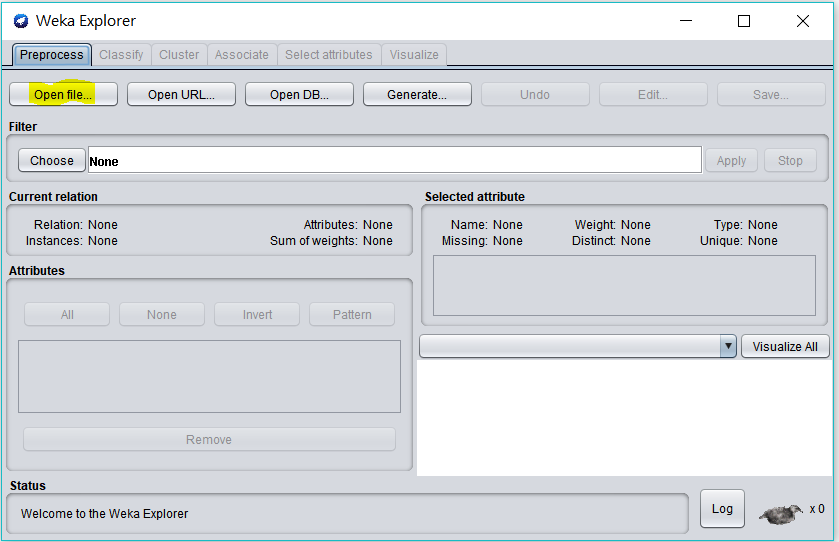
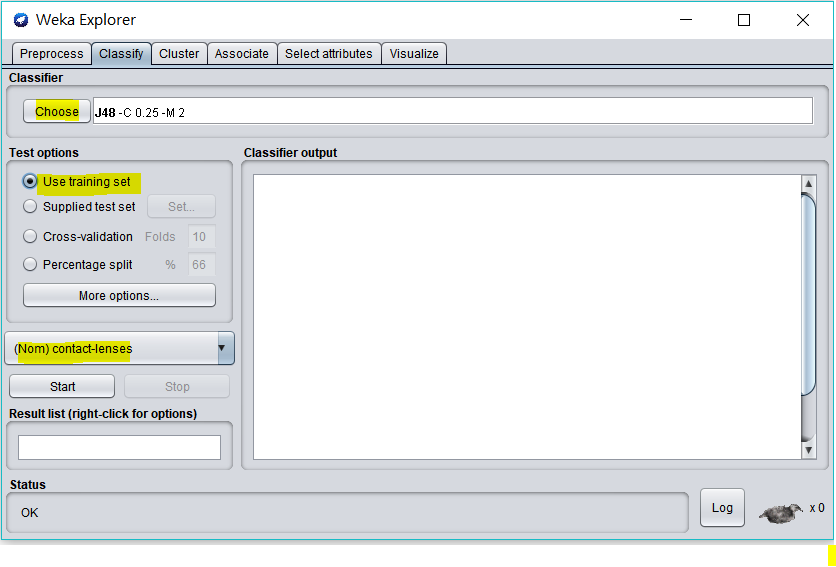
步驟

1. 點選「Open file」，選擇檔案「contact-lenses.arff」



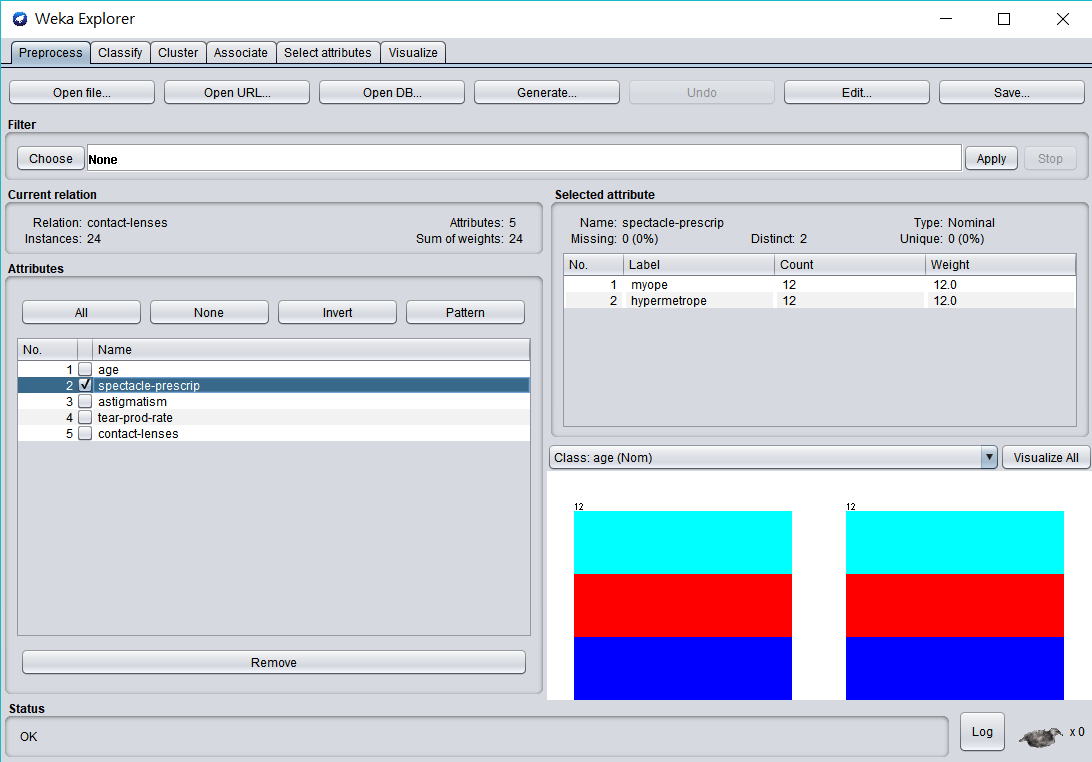
1. 在Classifier點選「Choose」，選擇weka/classifiers/trees/J48；在Test options選擇「Use training set」，；在Start上方的欄位選擇「(Nom)contact-lenses」。



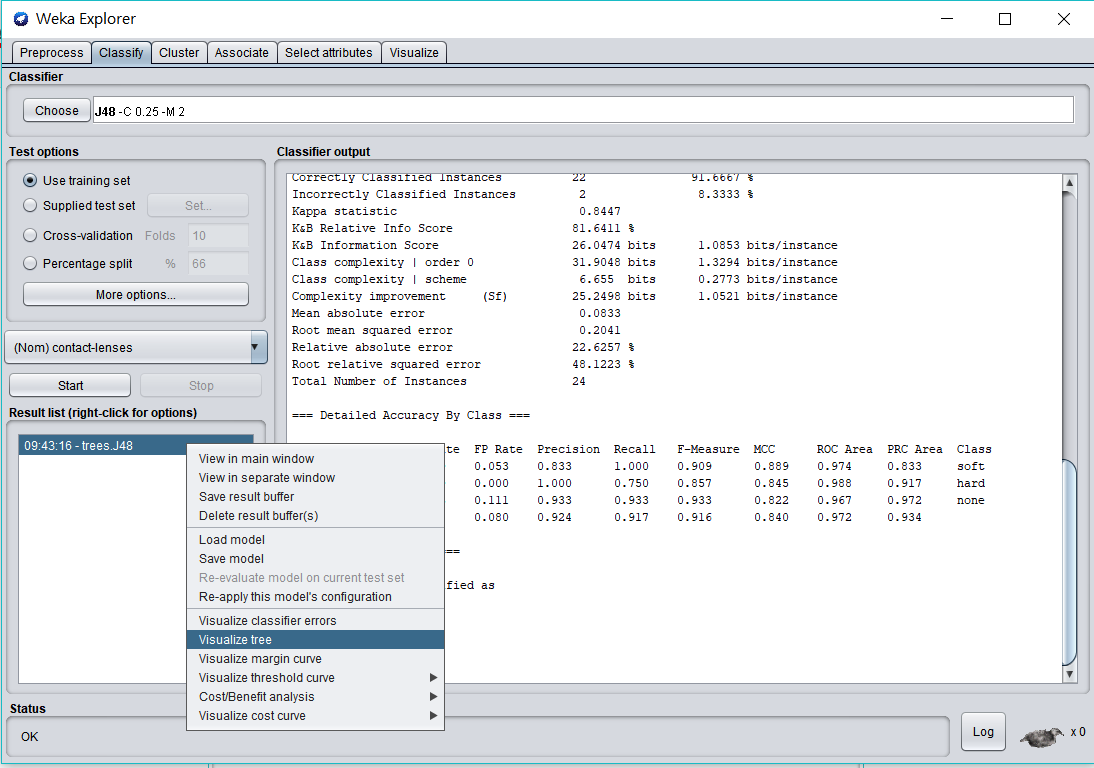
1. Attribute屬性選擇了spectacle-prescrip（包含了myope, hypermetrope）

Class選擇了age（屬性值包含young, pre-presbyopic, presbyopic）得到的直條圖則代表myope中young, pre-presbyopic, presbyopic各出現了多少次、hypermetrope中young, pre-presbyopic, presbyopic各出現了多少次。

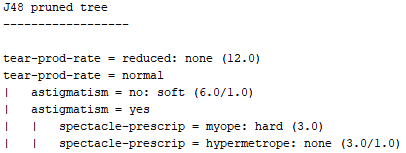
因此，長條圖的數量由Attribute決定、上方的數字代表Attribute中的各個屬性總共出現的次數、顏色種類由Class中的屬性值總數決定，每一種顏色代表Class中的一種屬性值。

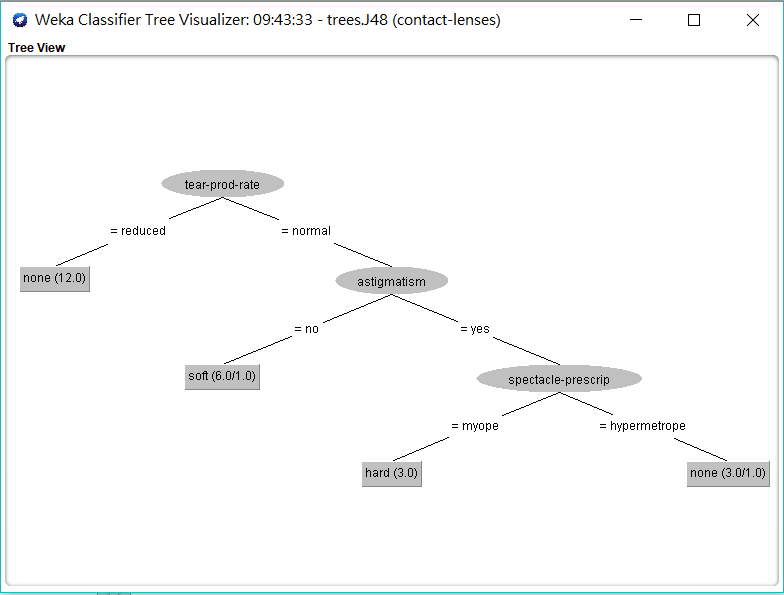


1. 在Result list選取Result，右鍵點選「Visualize tree」，即可看到決策樹。

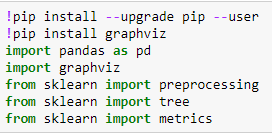


Tree 中的每一個節點 node（橢圓形），代表著不同的屬性測試，而最下方的葉 leaf（矩形），則代表分類的結果。由圖可以知道第一個Classification Rule是由tear-prod-rate屬於reduced還是normal分類，第二個Classification Rule是由astigmatism屬於no還是yes分類，第三個Classification Rule是由spectacle-prescrip屬於myope還是hypermetrope分類。





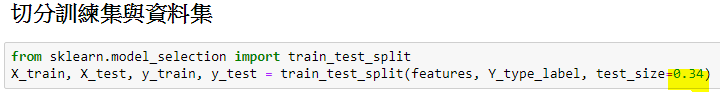
2. Python部分：
   * + 1. 新增指令使用套件「graphviz」



* + - 1. 讀取glass.csv檔案、設定x:input與y:output、將type轉為數字label、將x屬性合併成list

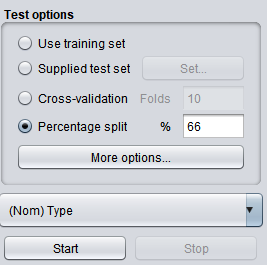


* + - 1. 設定測試集為0.34(34%)



Weka部分：

1. 點選「Open file」，選擇檔案「glass.arff」並在Classifier點選「Choose」，選擇weka/classifiers/trees/J48。
2. 選取「Percentage split」，輸入「66」。



1. Python部分：

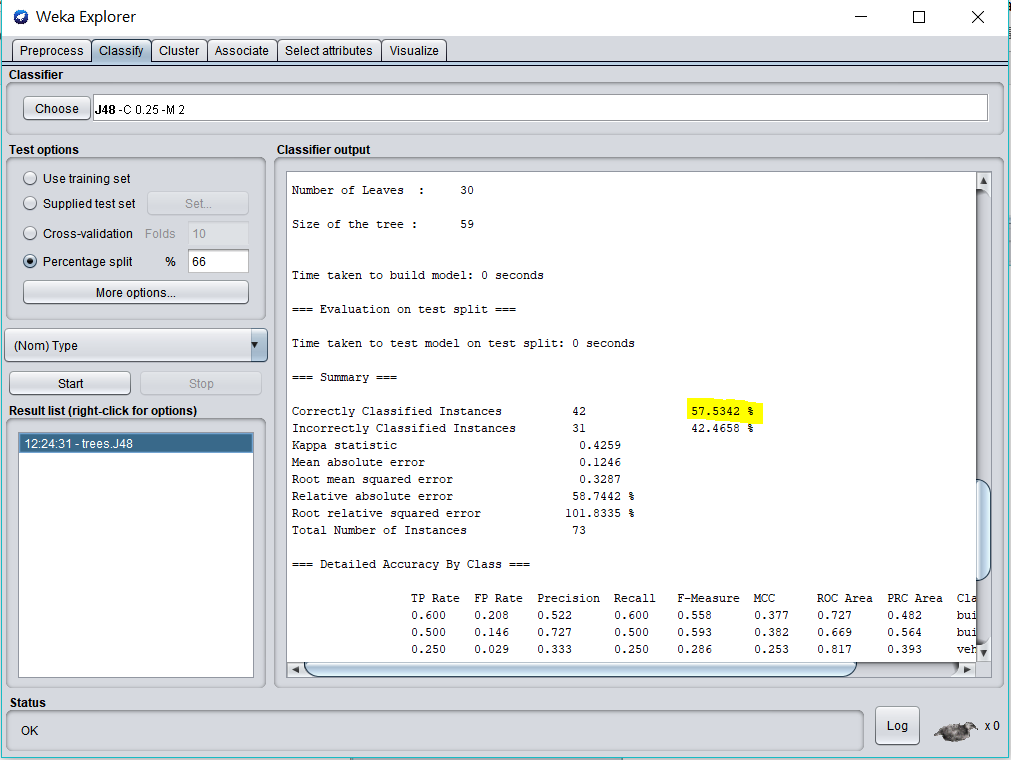
設定max\_depth=3, max\_leaf\_nodes=4。

下圖為使用python輸出的測試集精準度及訓練集精準度。

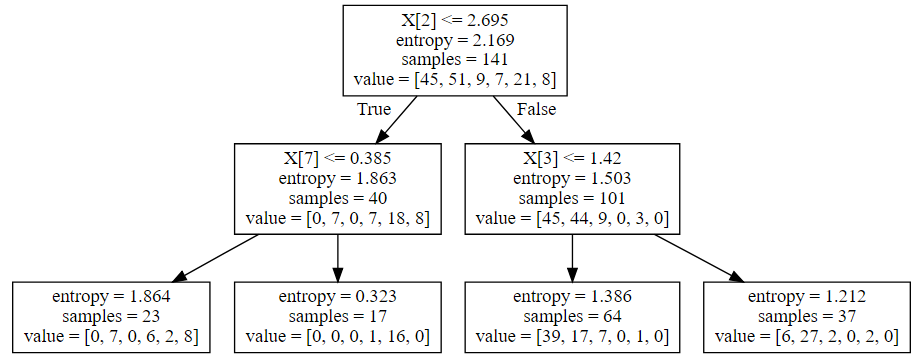


Weka部分：

下圖為Weka輸出的精準度。



比較兩圖可以發現Python與Weka輸出的精準度有差異，原因在於Python所切出的測試集與訓練集與Weka並不相同，所以結果也會不一樣。

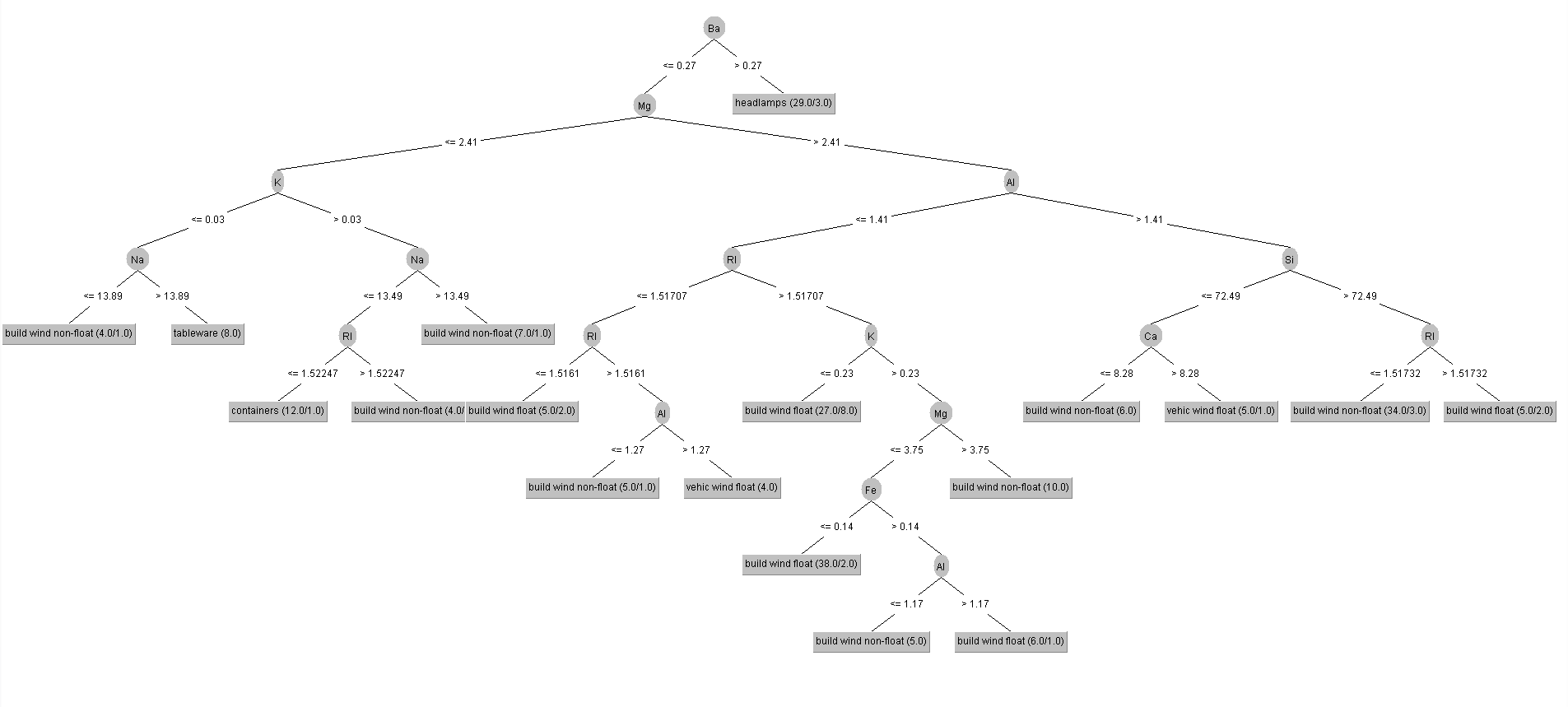


X[?] 表示使用哪些條件作為分類的區別。由 0-8 依序為

RI, Na, Mg, Al, Si, K, Ca, Ba, Fe。例如：第一層 X[2]<=2.169 表示 Mg 的量小於等於 2.169，若符合則歸類到True，不符合則歸類到 FALSE，以此類推。

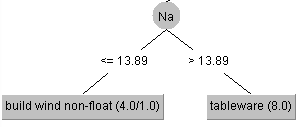
Sample表示樣本數，因為我們將測試集設為 66% 因此這邊的樣本數量為 214\*0.66 = 141。

Value 則是代表我們設定的 Attribute各自有多少個樣本，從左到右依序為build wind float、build wind non-float、vehic wind float、containers、headlamps、tableware。



Weka 跑出來的決策樹可以很直覺的知道分類的規則及依據，如：節點的標籤（橢圓形）表示已什麼做為資料的劃分的依據，而線上的判斷式則提供了分類條件。而方框的文字則代表 Output Attribute，方框內的括弧內的數字左邊表示符合的數量，右邊則表示不符合的數量。

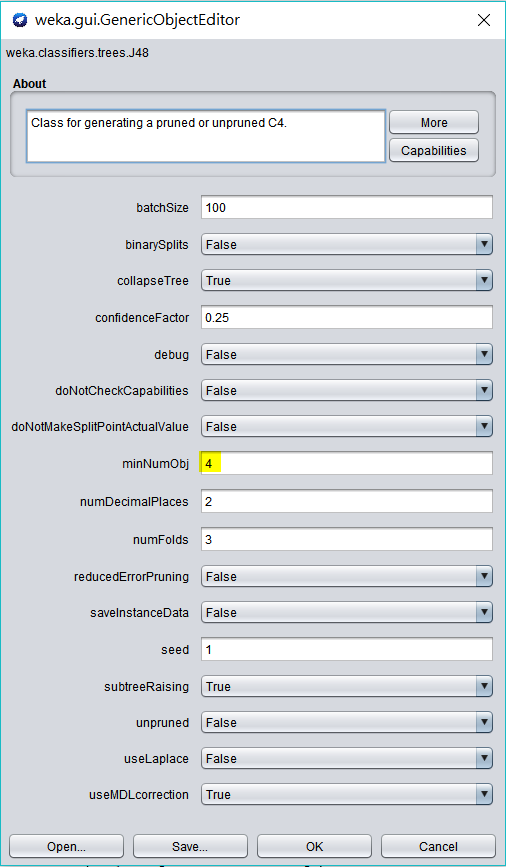
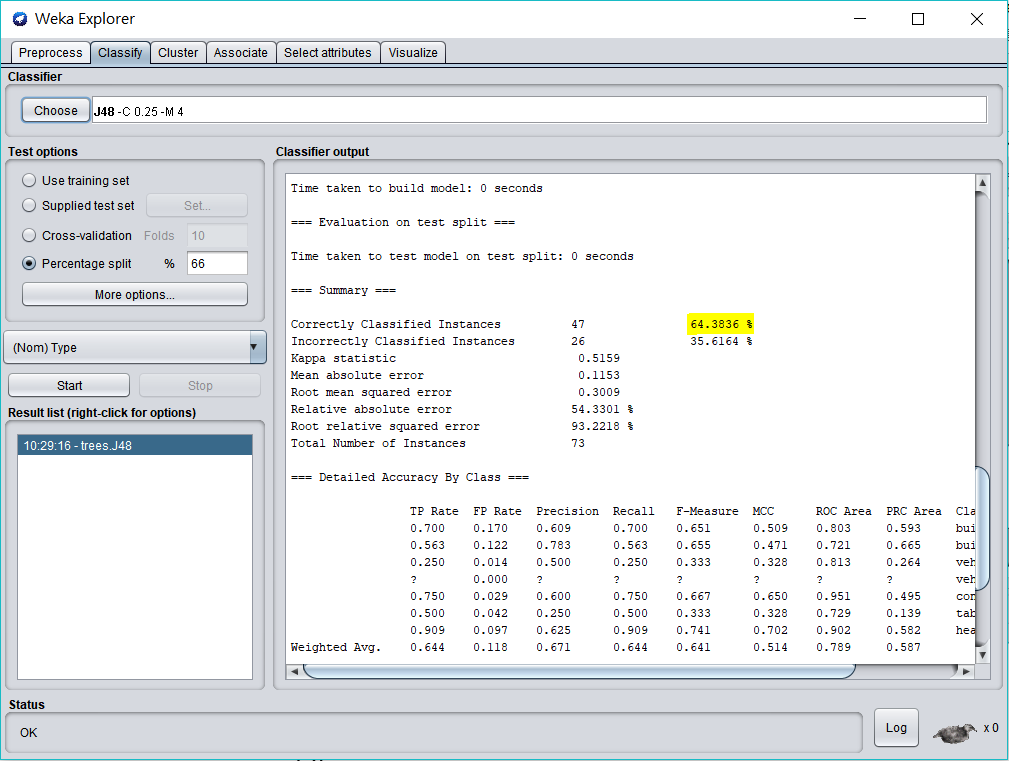
例如：Na當作分類的依據，若Na小於或等於13.89，則歸類到build wind non-float，而有4筆是歸類正確，1筆是歸類錯誤；若Na大於13.89，則歸類到tableware，而全部8筆都是歸類正確。



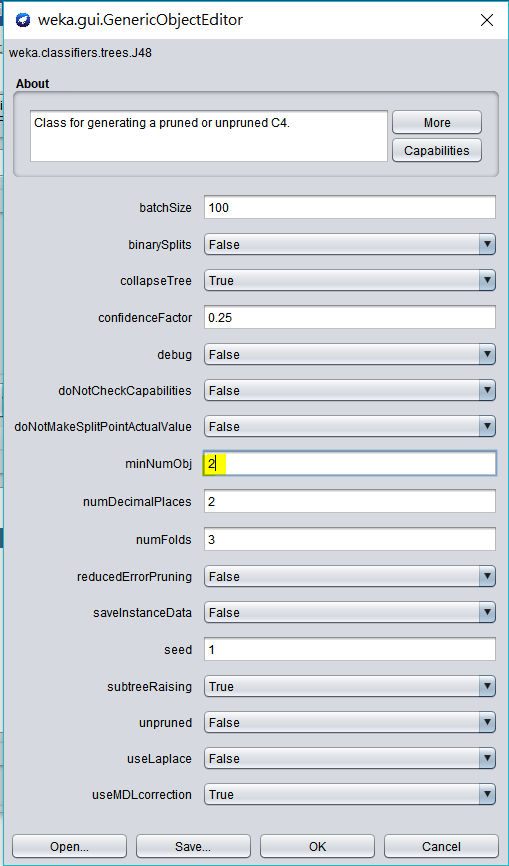
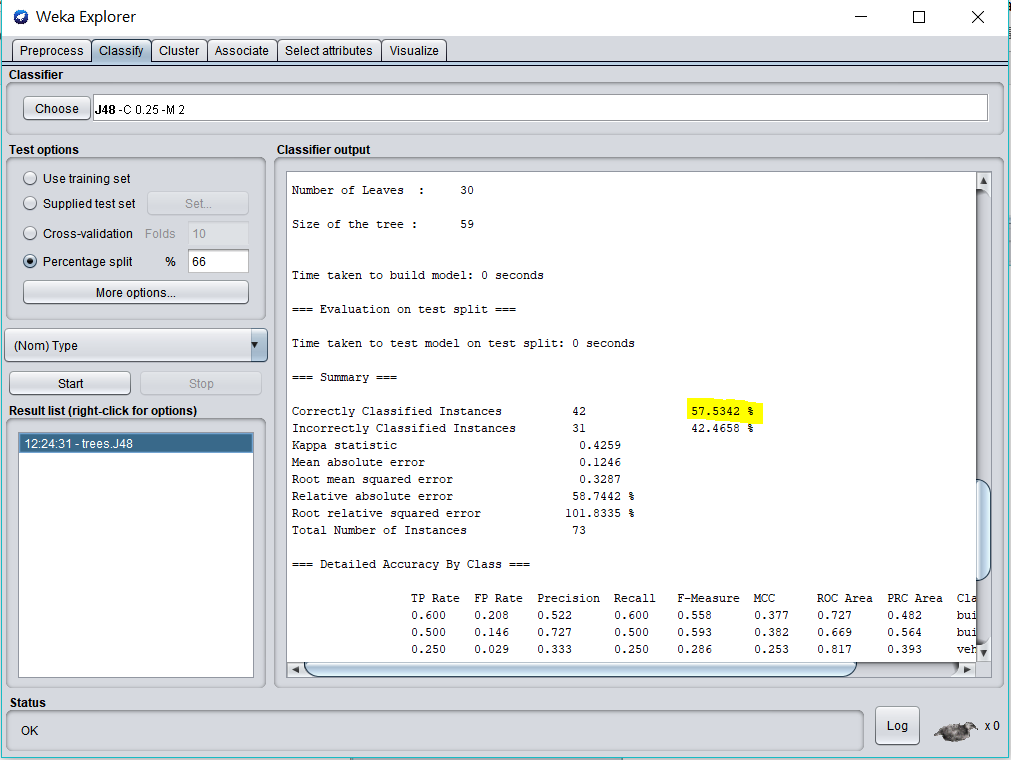
1. 首先比較參數值minNumObj的差異，測試集皆定為66%

點選「Choose」右方的欄位進行參數設定。

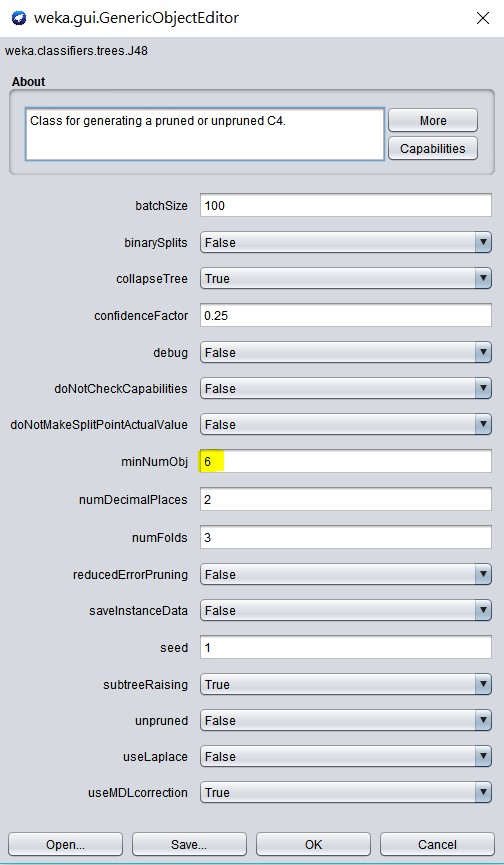
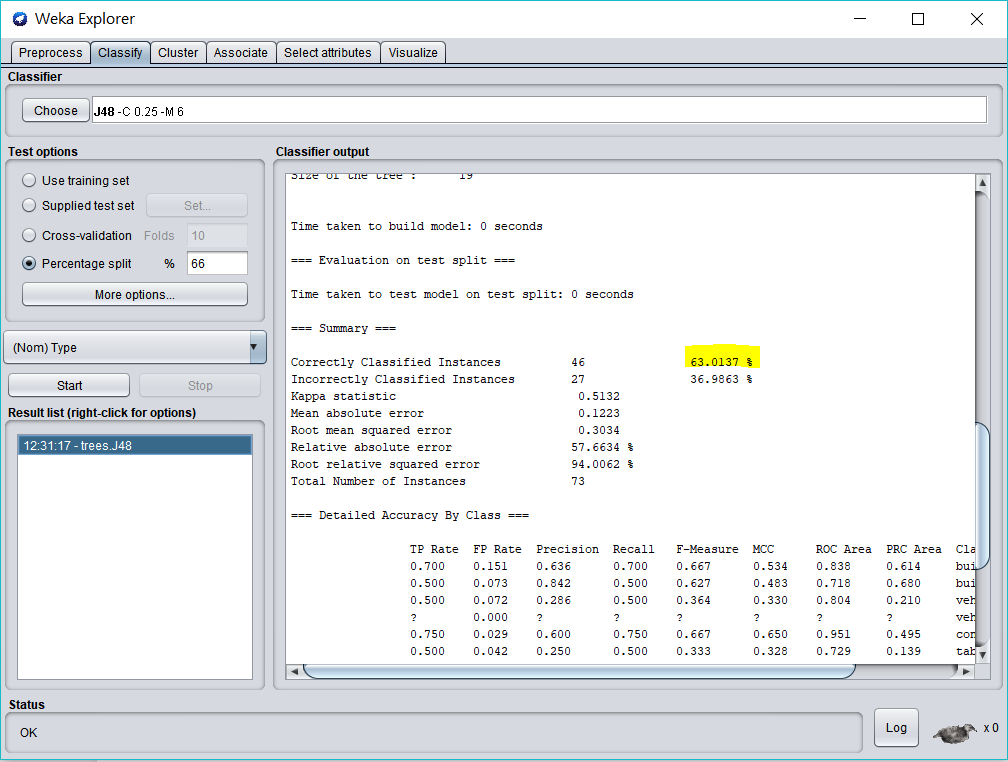
將minNumObj修改成4，得精準度為64.3836%。

將minNumObj修改成2，得精準度為57.5342%。

將minNumObj修改成6，得精準度為63.0137%。

由上兩圖及多次測試後可得minNumObj在等於4時，會有最大精準值，原因在於若minNumObj太小，會使模型複雜度越高則容易overfitting；但若minNumObj太大，因為資料數不足，反而會使精準度降低。

接著修改Percentage split參數，且固定minNumObj = 4

當Percentage split = 50%，準確度為64.486%



當Percentage split = 55%，準確度為55.2083%



當Percentage split = 60%，準確度為65.1163%



當Percentage split = 65%，準確度為61.3333%



當Percentage split = 70%，準確度為60.9375%



當Percentage split = 75%，準確度為62.2642%



當Percentage split = 80%，準確度為62.7907%



觀察趨勢後發現接近Percentage split = 60% 與 Percentage split = 50% 時可能會有最大準確度。因此，代入Percentage split = 58%, 59%, 61%, 62%, 48%, 49%,51%, 52% 進行測試。

得當Percentage split = 59% 時，會有最大準確度65.9091%



準確度會忽大忽小的原因在於，當Percentage split 很大時，會有更大的可能挑到有差異的部分；當Percentage split 很小時，則會有相對較小的可能挑到準確的部分。

訓練集的準確度通常是訓練數據上應用模型時獲得的準確度，而訓練集的準確度通常較測試集的準確度高是因為overfitting，此時的分類模型是不可靠的。