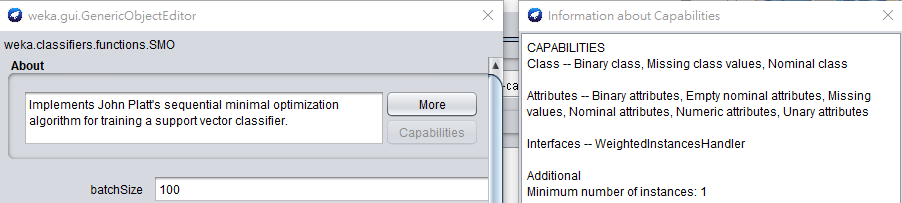
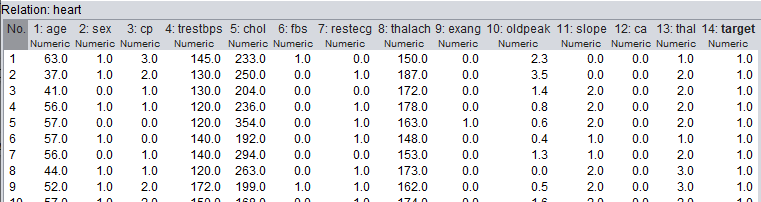
ECT HW4

**Weka部分:**

**(a)**



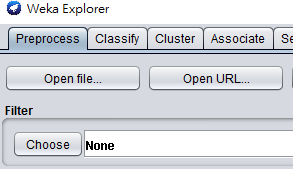
* 首先，去看看Weka SMO的使用條件，需要何種資料型態



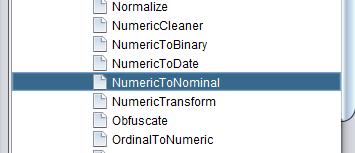
* 再來，去觀察看看目前的資料型態。

無法使用的原因 :

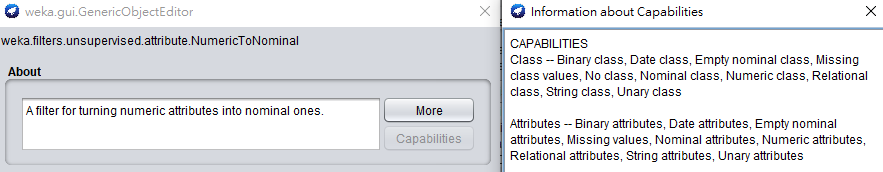
* 對比兩張圖後可以看出，Output不能接受Numeric的參數，但是我們的資料中所有型態皆是Numeric，因此我們要把Output的Target屬性改成Nominal。



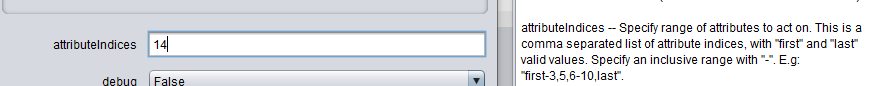
* 在Weka的Preprocess中找到Filter



* 在unsupervised 🡪 attribute中有一個Numeric轉Nominal的方法



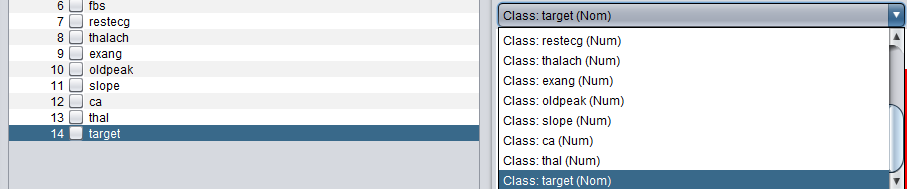
* 觀看使用條件，他幾乎甚麼都可以轉，連空值都行(雖然target沒有空值)，因此可以使用。



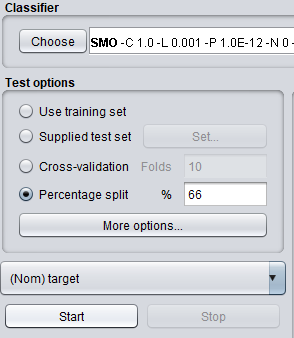
* 查看參數意義，可知道這一參數為指定attribute，因target為第14個屬性，因此填入14



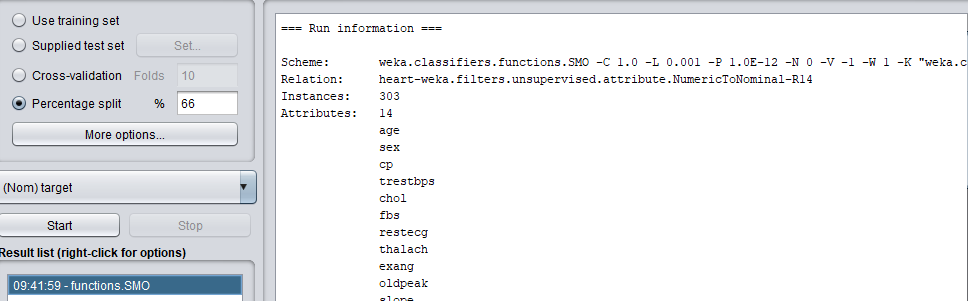
* 記得點選Apply，不然不會生效。這些操作都是暫存的，原始檔不會修改，想要修改記得另存新檔。



* 可以看到target屬性已經變為Nominal

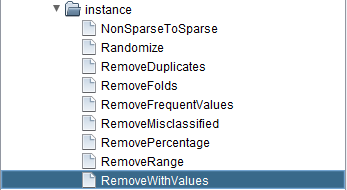


* 回去查看SMO的「Start」Button已經可以使用了

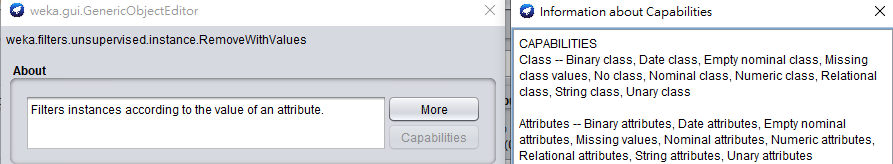


* 這裡雖然沒有處理missing data，但Weka的SMO已經可以分析了。可以注意Instance還是303，代表沒有刪除任何擁有missing data的實體。

**(b)**



* 因為要移除空值的資料，因此這次要去「Instance」中找方法使用



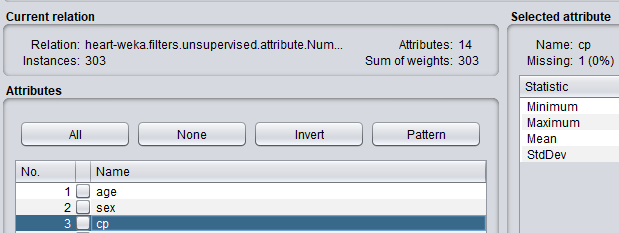
* 一樣先去看使用條件





* 然後再去看參數意義，這裡有2個參數特別重要 :「matchMissingValues」代表要match到missing data才刪除。

「attributeIndex」代表處理指定的屬性，這裡真的是「index」而不是「indices」，對!他是單數……所以你要一個一個去選擇屬性來處理。

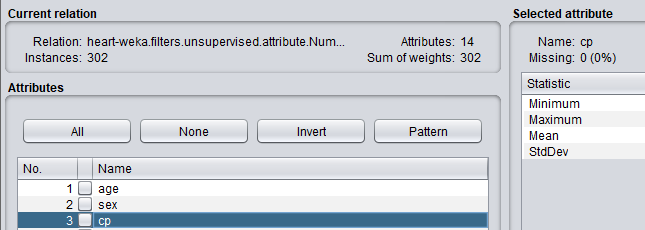


* 舉例來說，cp屬性有1個missing data，他是第3個屬性



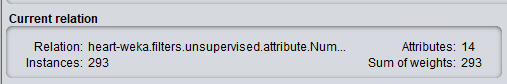


* Match設為true，Index設為3



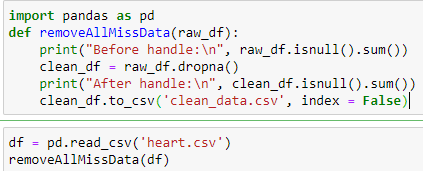
* 處理完後可看到missing變成0了，總Instance也少一個了(303-1) = 302

重複這個動作直到處理完所有資料，



* 處理完後會剩下293筆資料。

這是Weka的作法。我有另外寫了一個Python的小程序來處理。



* 用dropa()來處理空值，用to\_csv()來輸出檔案，index = False記得要設，不然Weka無法讀取，因為他會多一個column。

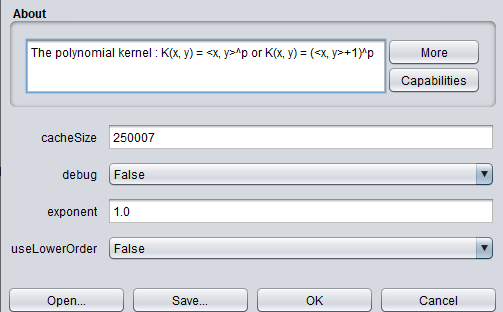
 

* 並且可看到這種輸出結果。

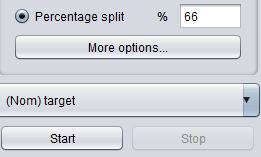
**(c)**



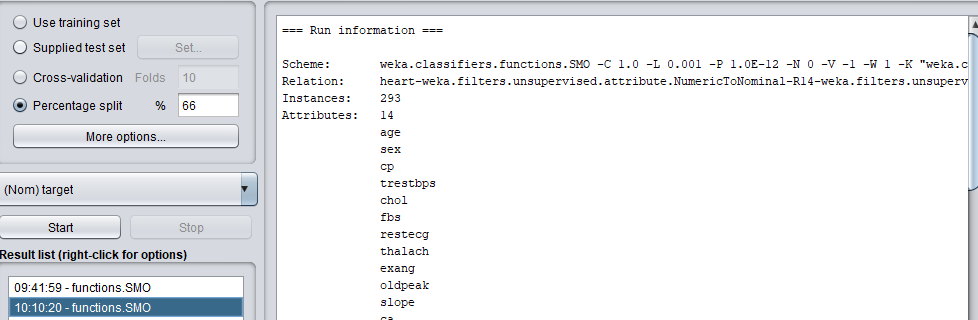
* 先到SMO的kernel部分，選擇好PolyKernal，這個方法可以決定要用幾次方的Kernel來訓練。



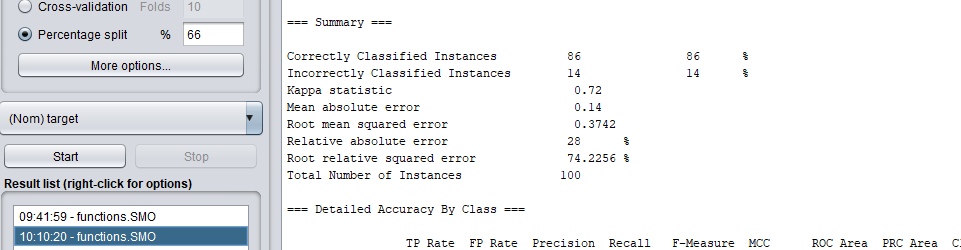
* 因題目要求「Linear」，所以exponent設為1



* 依照題目要求，Percentage spilt設為66%。



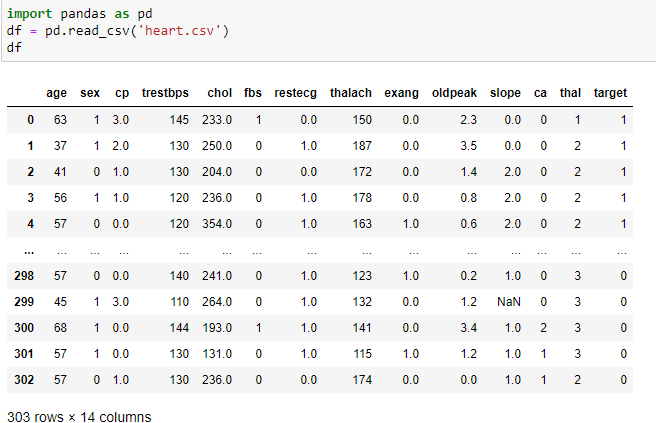
* 訓練後可看到這次並非用303筆資料，而是203筆資料在訓練。



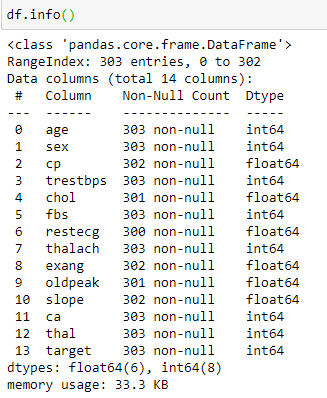
* 準確率為86%，比未處理missing data時的83%還準，因此可判斷此步驟是有一定程度的意義。

**Python部分 :**

**(d)**



* 先導入資料



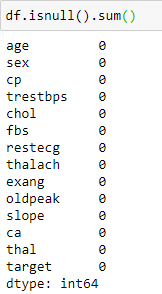
* Info()函數可以看到各種資料，可看出總共有303筆資料，但有6種屬性的資料non-null的數量並非303筆，因此可推斷有missing data。



* 使用isnull().sum()直接統整出有多少筆missing data，在此為10筆。

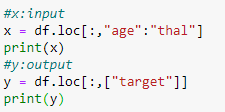


* 使用dropnd()函數來去除missing data，記住要重新，不然DataFrame並不會改變。

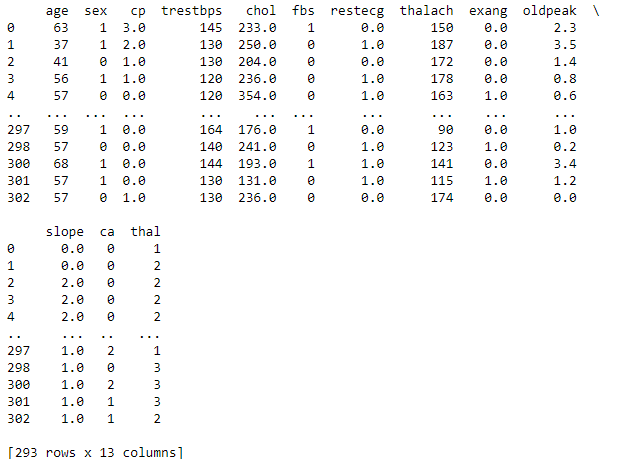


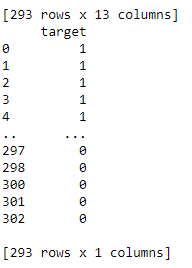
* 在觀察一次可看出missing data已經沒有了

**(e)**

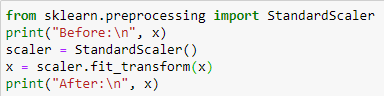


* 用loc函數切分為input、output。

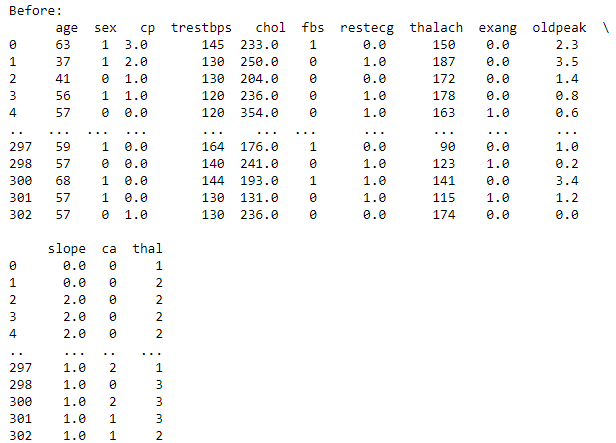


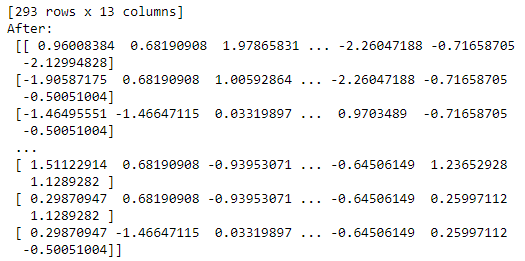


**(f)**



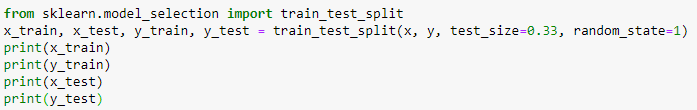
* 我在這裡先有印出來看原始狀態，在用fit\_transform()轉換後assign回給x，並且再次印出來檢查是否有轉換成功。結果如下:





* 看的出來已經轉為標準化的形式了。都以0為中心點左右偏移。

**(g)**



* 依照題目要求，用train\_test\_split()切分為training、testing data。

「最前面的兩個參數x, y」 : 代表我們要切割的資料

「test\_size=0.33」: 代表testing data佔33%

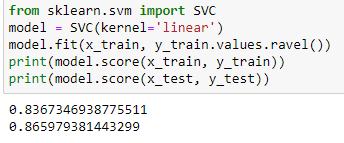
「random\_state=1」: 為了讓他每次切的結果都一樣，可以想像成它是一個seed，每次都基於這個來產生相同的結果。1會有1的結果、2也會有2個結果。若要每次切的結果不同可以不要設定這個參數。(P.S. 這是為了評分時大家答案一樣用的吧)





* 印出來可以看到他們被切割為training : 196、testing : 97。驗算一下293\*0.33 = 96.69 ≒ 97。結果是正確的。

**(h)**



* 先import SVC，並在用SVC()建立model時設定kernal = linear。之後就把data fit進去訓練。用score函數分析準確度，因題目只說分析準確度，所以我把training、testing data都拿進去分析了。但真正重要的是test。