Project #2 Report

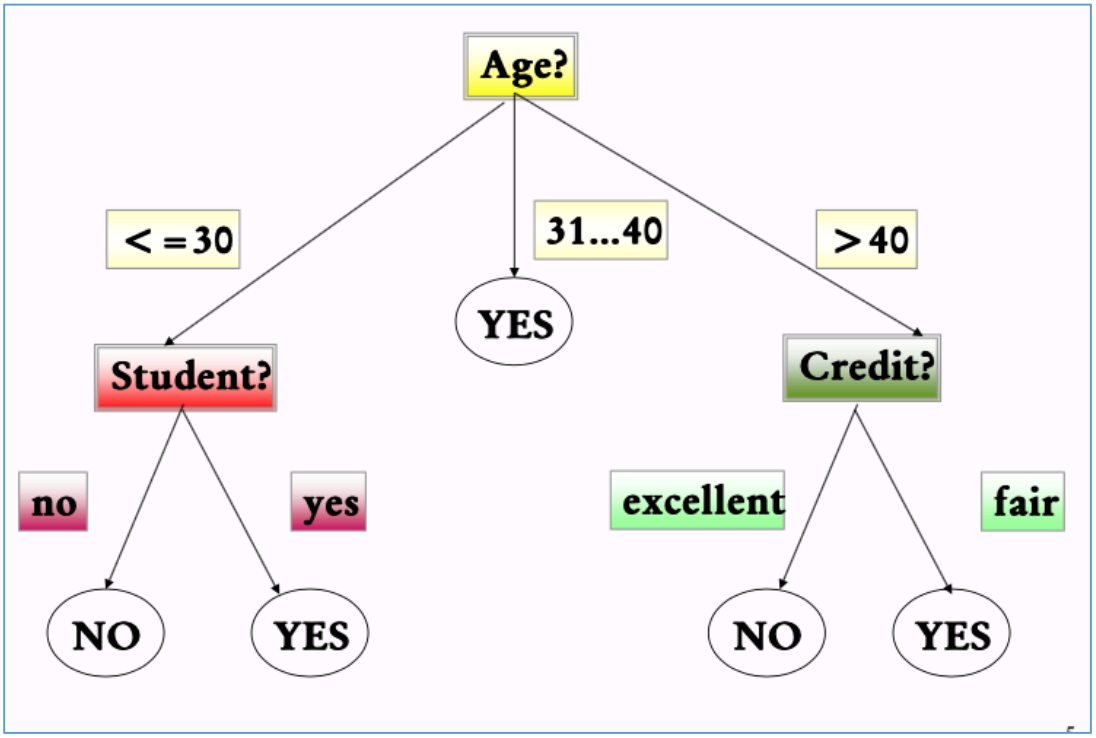
名字：王康霖
學號：N26077124

執行指令：python3 predict.py

Dataset

問題設計：客戶是否會買電腦

- My Decision Tree Rules



這是一個結果為“會/不會買電腦”的 Decision Tree
依據上面的 Decision Tree 產生下列有 30 筆 Data 的 Dataset
因此在這個問題上
M = 30, K = 4

- Dataset

Age	Income	Student	Credit	Buys_co...
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
31...40	medium	yes	fair	yes
>40	high	no	fair	yes
<=30	medium	no	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
>40	high	yes	excellent	no
<=30	medium	yes	fair	yes
>40	high	no	excellent	no
<=30	medium	yes	excellent	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
<=30	low	yes	fair	yes
>40	low	no	fair	yes
31...40	low	no	fair	yes
>40	medium	yes	fair	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no
31...40	medium	no	excellent	yes
<=30	high	yes	excellent	yes
31...40	low	yes	fair	yes
>40	low	no	excellent	yes
31...40	high	no	excellent	yes
<=30	low	no	excellent	no
<=30	low	yes	excellent	yes
31...40	medium	yes	excellent	yes
>40	medium	yes	excellent	no

Data Preprocessing

因為 Dataset 裡面的資料是以文字形式在儲存，但這樣無法拿來做使用。所以我們需要先用 label encode 的方式將同組轉換成數字。例如把 `Age` 欄位中的 '<=30', '31...40', '>40' 分別轉換成 1，2，3。我是使用 sklearn.preprocessing 所提供的 LabelEncoder。

再來就是要將 Feature 和 Answer 分開，Feature 就是前四個欄位：`Age`, `Income`, `Student`, `Credit`。Answer 就是最後的欄位 `Buys_computer`。
最後再將 dataset split 成 train_data 和 test_data，使用的是 sklearn.model_selection 所提供的 train_test_split 功能。
參數如下：

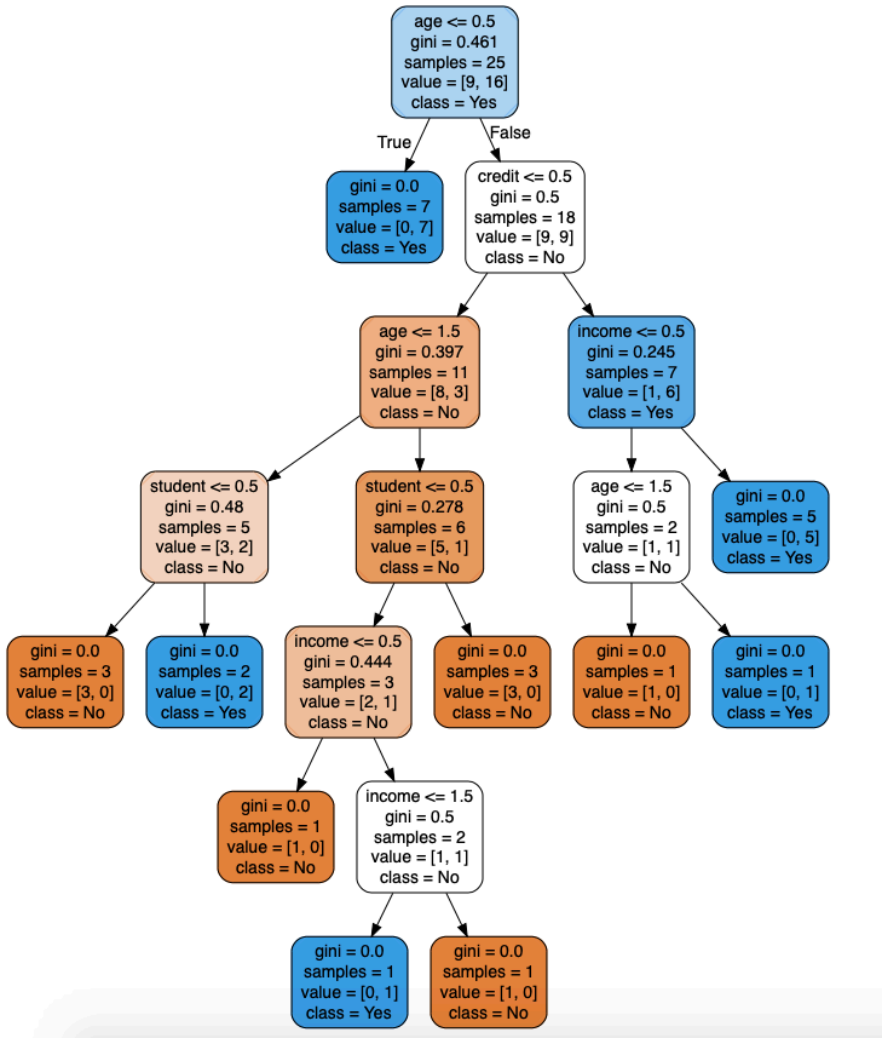
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.15,
random_state=42)
```

random_state 會設 42 而不是亂數是為了讓 data 可以一樣

Decision Tree

- 結果

使用的是 sklearn 所提供的套件 tree，數的生成是將規則輸出到 `dot_tree.txt` 再用線上工具來繪畫（Webgraphiz, http://webgraphviz.com）



這個 Decision Tree 的準確率為 0.8

```
Answer:['1' '1' '1' '1' '0']

Decision Tree
Predict: ['1' '1' '1' '1' '1']
Accuracy: 0.8
```

- 討論

其實在 random_state 為其他數字的時候，也就是分割檔案的方式不一樣的時候所得到的準確率都有上下，所以這個分割參數還不是最佳的參數，但為了顯示差異所以選擇這個 random_state。

另外，由於 dataset 的 M 只有 30，資料其實是非常不足的，而且由於 dataset 中沒有重複的資料，所以在 split_data 的時候有可能會發生 model 從來沒學過的規則出現在 test_data 中，所以給出錯誤的答案其實也是很合理的。

SVM

- 結果

```
SVM
Predict: ['1' '1' '1' '1' '1']
Accuracy: 0.8
```

可以看到準確率也是 0.8，但其實這也是在 train_split_test 的時候選擇的 random_state 的關係而已。在其他的分割方法的時候，準確率可能是 SVM > Decision_Tree 亦或是反過來。

總結

在 K 和 M 都相同的情況下，我們可以發現會根據我們的 Dataset 的 train 和 test 的不同而有不同的結果。得出的結論是 Decision Tree 結構簡單，使用上也不需要設定參數，但缺點就是“想像力”太差，如果出現的分類是自己沒有看過的，打錯的機率就很高。而其他分類器的方法，例如本次所用的 SVM，因為還有很多參數可以做調整，而且資料集只有 30 筆資料，沒有辦法做出顯著的效果，並且也實在無法去跟 Decision Tree 來做相提並論和比較。