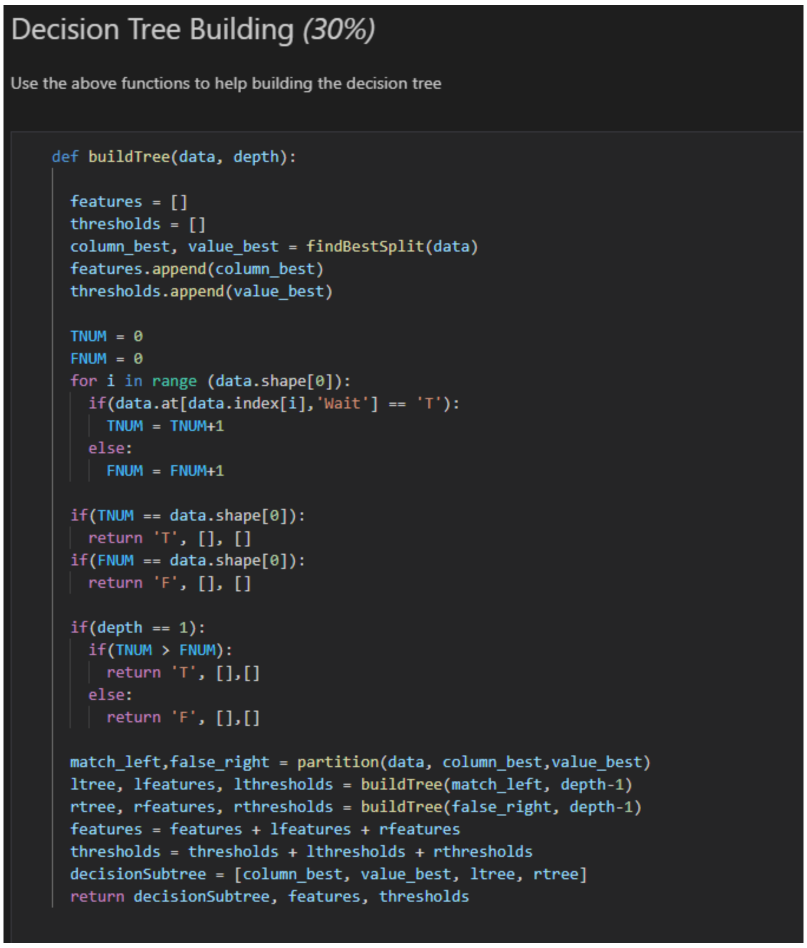


Decision Tree Building *(30%)*

Use the above functions to help building the decision tree

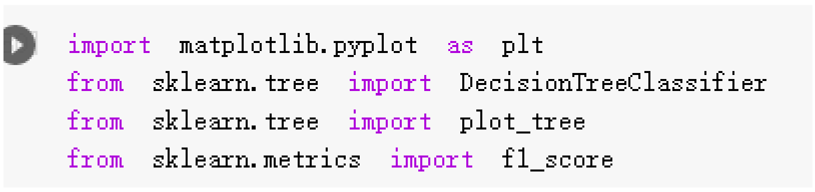
```python
def buildTree(data, depth):

    features = []
    thresholds = []
    column_best, value_best = findBestSplit(data)
    features.append(column_best)
    thresholds.append(value_best)

    TNUM = 0
    FNUM = 0
    for i in range (data.shape[0]):
      if(data.at[data.index[i],'Wait'] == 'T'):
        TNUM = TNUM+1
      else:
        FNUM = FNUM+1

    if(TNUM == data.shape[0]):
      return 'T', [], []
    if(FNUM == data.shape[0]):
      return 'F', [], []

    if(depth == 1):
      if(TNUM > FNUM):
        return 'T', [],[]
      else:
        return 'F', [],[]

    match_left,false_right = partition(data, column_best,value_best)
    ltree, lfeatures, lthresholds = buildTree(match_left, depth-1)
    rtree, rfeatures, rthresholds = buildTree(false_right, depth-1)
    features = features + lfeatures + rfeatures
    thresholds = thresholds + lthresholds + rthresholds
    decisionSubtree = [column_best, value_best, ltree, rtree]
    return decisionSubtree, features, thresholds
```

先計算此 node 的 Wait 有多少個 True 跟 False。
如果 True 或 False 已經達到此 node 的 row 數量，就代表已經不用繼續分下去了，直接 return。如果 depth 已經到最後一層了也直接 return True 或 False 比較多的那方。

如果都不成立，則開始造樹，用 partition 做出 match branch 與 not match function，並再次呼叫 buildstree 來做出 subtree, next_features, next_threshold，形成一個遞迴。
最後 features +後來使用的 next_features，thresholds+後來使用的 next_thresholds，然後 return。

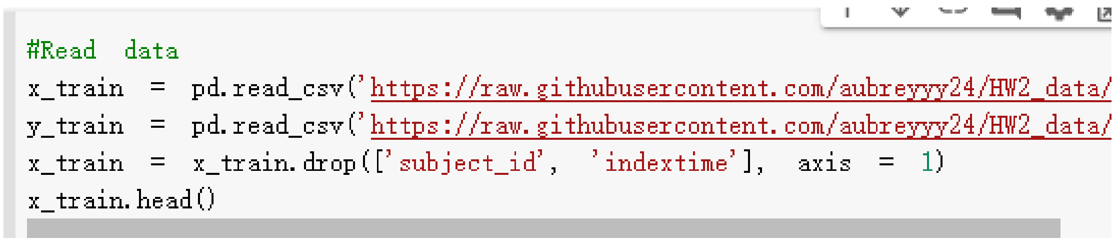
Top 3 splitting features and their thresholds of model:

CMO，MVAR12，MVAR23



```
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
from sklearn.metrics import f1_score
```

Import package 來輔助建樹
F1_score 用來評量 model



```
#Read data
x_train = pd.read_csv('https://raw.githubusercontent.com/aubreyyy24/HW2_data/
y_train = pd.read_csv('https://raw.githubusercontent.com/aubreyyy24/HW2_data/
x_train = x_train.drop(['subject_id', 'indextime'], axis = 1)
x_train.head()
```

Drop 掉無法處理的資料型態。

用 cpp pruning 剪掉幫助較少的枝，避免 overfit。
Alpha 是用 for loop 跑完的結果，最適合的值。
剩下就是利用 model 來 predict。