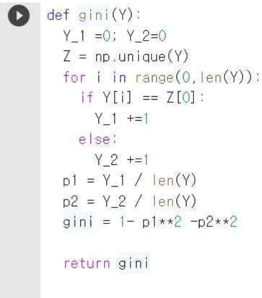
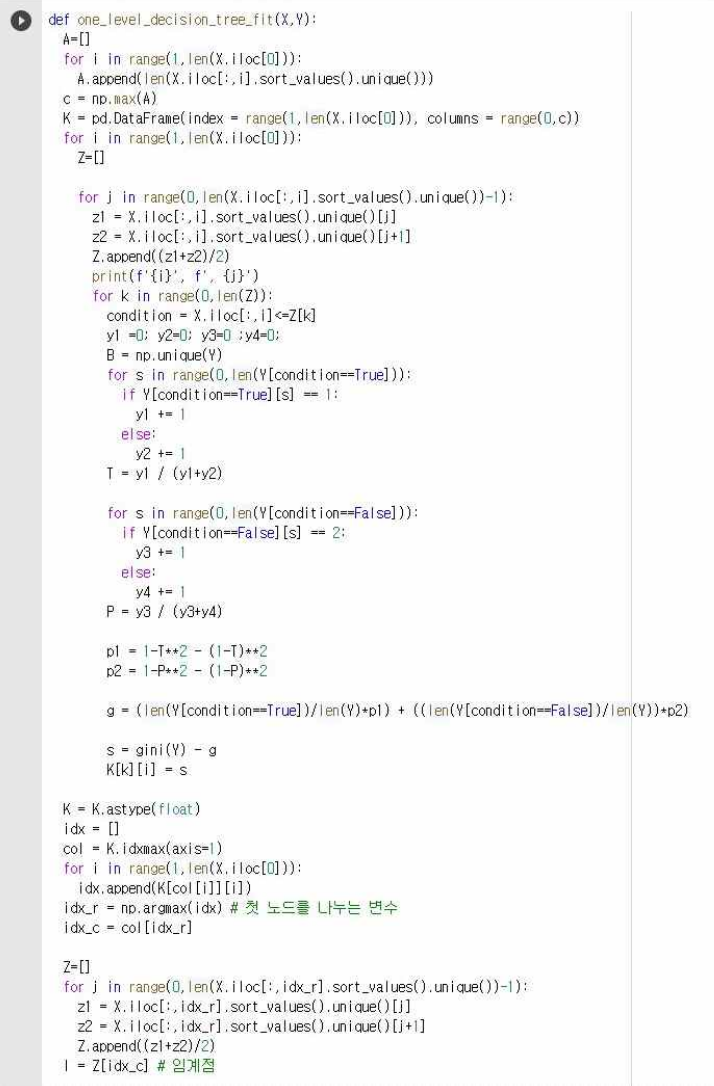
HW6

2021321148
INSIK CHO



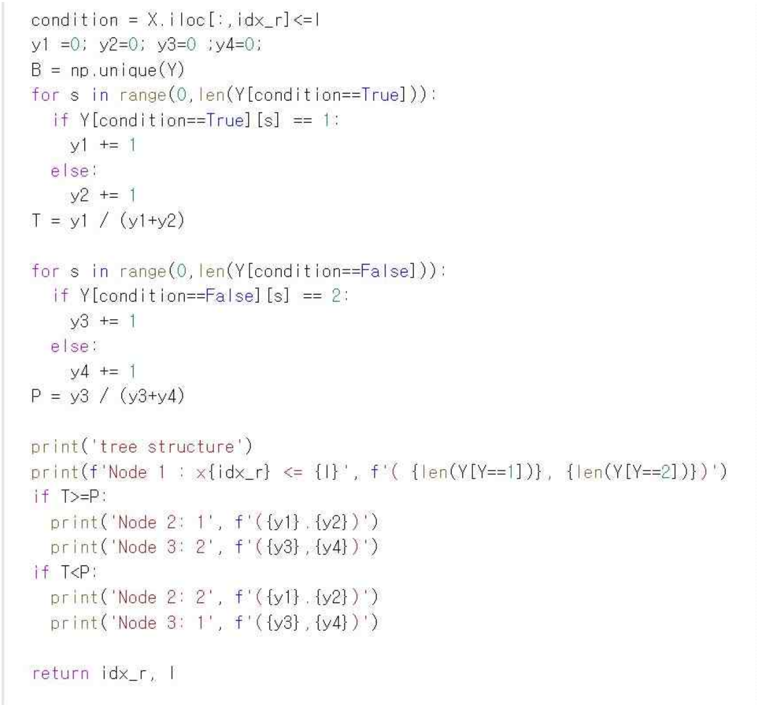
```
def gini(Y):
  Y_1 =0; Y_2=0
  Z = np.unique(Y)
  for i in range(0,len(Y)):
    if Y[i] == Z[0]:
      Y_1 +=1
    else:
      Y_2 +=1
  p1 = Y_1 / len(Y)
  p2 = Y_2 / len(Y)
  gini = 1- p1**2 -p2**2

  return gini
```

처음엔 지니계수를 정의했습니다.



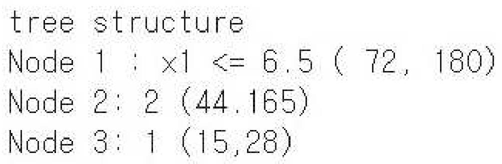
```
def one_level_decision_tree_fit(X,Y):
  A=[]
  for i in range(1,len(X.iloc[0])):
    A.append(len(X.iloc[:,i].sort_values().unique()))
  c = np.max(A)
  K = pd.DataFrame(index = range(1,len(X.iloc[0])), columns = range(0,c))
  for i in range(1,len(X.iloc[0])):
    Z=[]

    for j in range(0,len(X.iloc[:,i].sort_values().unique())-1):
      z1 = X.iloc[:,i].sort_values().unique()[j]
      z2 = X.iloc[:,i].sort_values().unique()[j+1]
      Z.append((z1+z2)/2)
      print(f'{i}', f', {j}')
      for k in range(0,len(Z)):
        condition = X.iloc[:,i]<=Z[k]
        y1 =0; y2=0; y3=0 ;y4=0;
        B = np.unique(Y)
        for s in range(0,len(Y[condition==True])):
          if Y[condition==True][s] == 1:
            y1 += 1
          else:
            y2 += 1
        T = y1 / (y1+y2)

        for s in range(0,len(Y[condition==False])):
          if Y[condition==False][s] == 2:
            y3 += 1
          else:
            y4 += 1
        P = y3 / (y3+y4)

        p1 = 1-T**2 - (1-T)**2
        p2 = 1-P**2 - (1-P)**2

        g = (len(Y[condition==True])/len(Y)*p1) + ((len(Y[condition==False])/len(Y))*p2)

        s = gini(Y) - g
        K[k][i] = s

  K = K.astype(float)
  idx = []
  col = K.idxmax(axis=1)
  for i in range(1,len(X.iloc[0])):
    idx.append(K[col[i]][i])
  idx_r = np.argmax(idx) # 첫 노드를 나누는 변수
  idx_c = col[idx_r]

  Z=[]
  for j in range(0,len(X.iloc[:,idx_r].sort_values().unique())-1):
    z1 = X.iloc[:,idx_r].sort_values().unique()[j]
    z2 = X.iloc[:,idx_r].sort_values().unique()[j+1]
    Z.append((z1+z2)/2)
  l = Z[idx_c] # 임계점
```

위와 같은 코드를 작성해, 어떤 변수를 기준으로 처음에 분기하는지(idx_r), 그리고 수치가 몇인지(l)를 계산했습니다.



```
condition = X.iloc[:,idx_r]<=l
y1 =0; y2=0; y3=0 ;y4=0;
B = np.unique(Y)
for s in range(0,len(Y[condition==True])):
  if Y[condition==True][s] == 1:
    y1 += 1
  else:
    y2 += 1
T = y1 / (y1+y2)

for s in range(0,len(Y[condition==False])):
  if Y[condition==False][s] == 2:
    y3 += 1
  else:
    y4 += 1
P = y3 / (y3+y4)

print('tree structure')
print(f'Node 1 : x{idx_r} <= {l}', f'( {len(Y[Y==1])}, {len(Y[Y==2])})')
if T>=P:
  print('Node 2: 1', f'({y1}.{y2})')
  print('Node 3: 2', f'({y3},{y4})')
if T<P:
  print('Node 2: 2', f'({y1}.{y2})')
  print('Node 3: 1', f'({y3},{y4})')

return idx_r, l
```

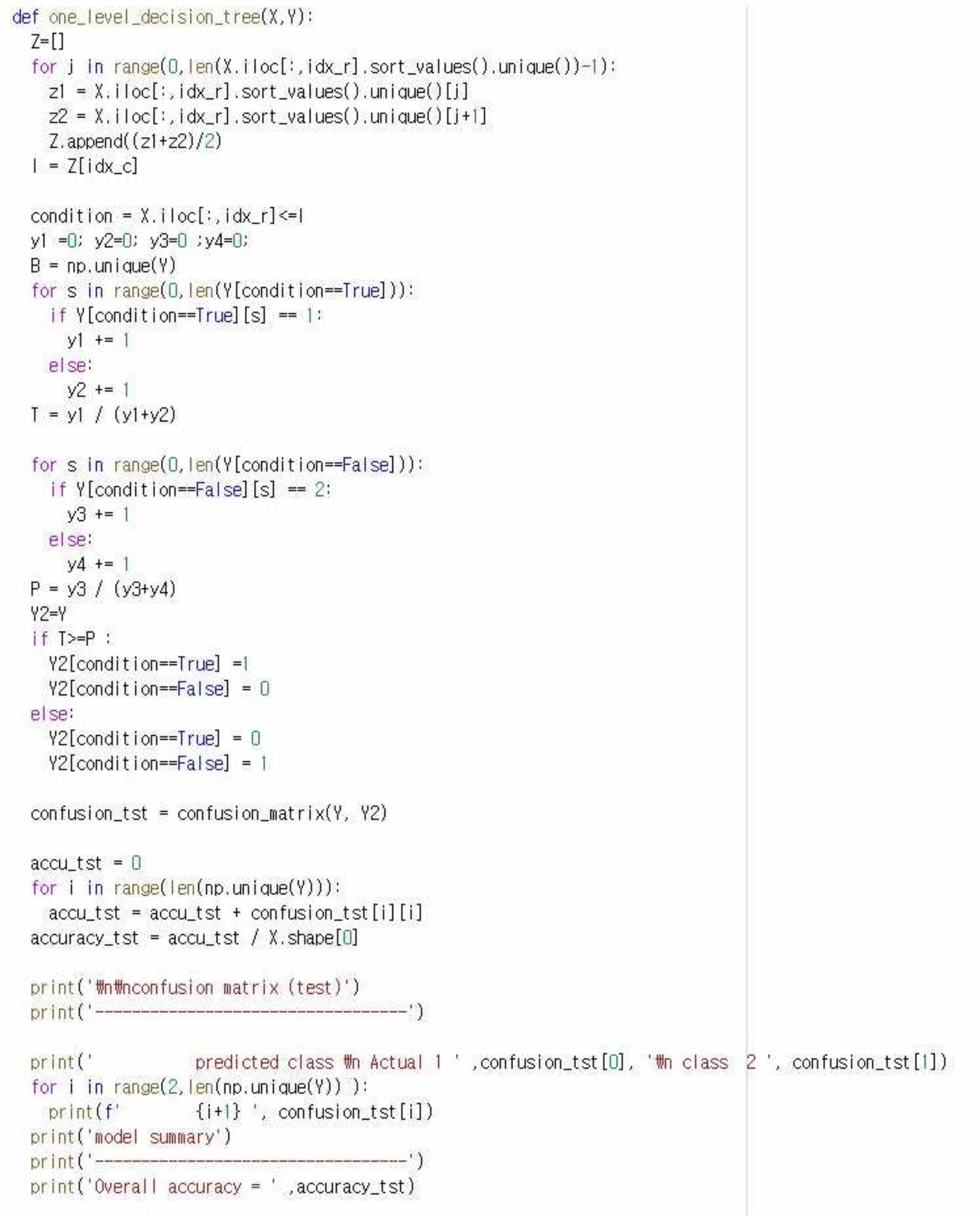
추가로 문제에 주어진 형태로 출력이 되도록 했습니다. 해당 값을 확인하면,



```
tree structure
Node 1 : x1 <= 6.5 ( 72, 180)
Node 2: 2 (44.165)
Node 3: 1 (15,28)
```

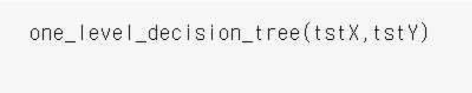
위와 같은 값이 나옵니다.

추가로 TST데이터를 검사하기 위한 코드는



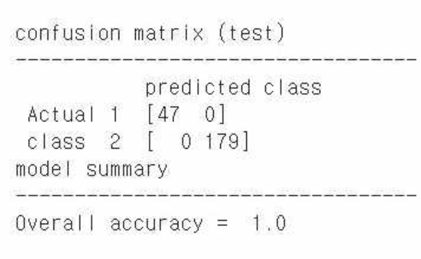
```
def one_level_decision_tree(X,Y):
  Z=[]
  for j in range(0,len(X.iloc[:,idx_r].sort_values().unique())-1):
    z1 = X.iloc[:,idx_r].sort_values().unique()[j]
    z2 = X.iloc[:,idx_r].sort_values().unique()[j+1]
    Z.append((z1+z2)/2)
  l = Z[idx_c]

  condition = X.iloc[:,idx_r]<=l
  y1 =0; y2=0; y3=0 ;y4=0;
  B = np.unique(Y)
  for s in range(0,len(Y[condition==True])):
    if Y[condition==True][s] == 1:
      y1 += 1
    else:
      y2 += 1
  T = y1 / (y1+y2)

  for s in range(0,len(Y[condition==False])):
    if Y[condition==False][s] == 2:
      y3 += 1
    else:
      y4 += 1
  P = y3 / (y3+y4)
  Y2=Y
  if T>=P :
    Y2[condition==True] =1
    Y2[condition==False] = 0
  else:
    Y2[condition==True] = 0
    Y2[condition==False] = 1

  confusion_tst = confusion_matrix(Y, Y2)

  accu_tst = 0
  for i in range(len(np.unique(Y))):
    accu_tst = accu_tst + confusion_tst[i][i]
  accuracy_tst = accu_tst / X.shape[0]

  print('₩n₩nconfusion matrix (test)')
  print('-----------------------------------')

  print('           predicted class ₩n Actual 1 ' ,confusion_tst[0], '₩n class  2 ', confusion_tst[1])
  for i in range(2,len(np.unique(Y)) ):
    print(f'        {i+1} ', confusion_tst[i])
  print('model summary')
  print('-----------------------------------')
  print('Overall accuracy = ' ,accuracy_tst)
```

위와 같이 작성하였으며, 예측을 진행한 결과



```
one_level_decision_tree(tstX,tstY)
```

좌측과 같은 수치를 얻었습니다. (1.0???)



```
confusion matrix (test)
-----------------------------------
           predicted class
 Actual 1  [47  0]
 class  2  [  0 179]
model summary
-----------------------------------
Overall accuracy =  1.0
```