**HW12**

201635825 오성원

**HW12**

|  |
| --- |
| Source Code |
| import pandas as pd  import numpy as np  import random  from math import pow  #Read the file  df = pd.read\_excel('C:/python\_file/HW12\_data.xlsx')  result=pd.DataFrame(index=range(0,12), columns=[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0])  def bubble\_sort(arr):  for i in range(len(arr[0]) - 1, 0, -1):  for j in range(i):  if arr[0][j] > arr[0][j + 1]:  arr[0][j], arr[0][j + 1] = arr[0][j + 1], arr[0][j]  arr[1][j], arr[1][j + 1] = arr[1][j + 1], arr[1][j]  def decision\_split\_value(arr):  bubble\_sort(arr)  min\_gini=1  split\_result=0  val=0  for i in range(11):  """\_0 is the number of y-values of -1,  and \_1 is the number of y-values of 1."""  high\_1=0  high\_0=0  low\_1=0  low\_0=0  gini=0  if (i==0):  split=(0+arr[0][i])/2  elif (0<i<=9):  if (arr[0][i-1]==arr[0][i]):  continue  else:  split=(arr[0][i-1]+arr[0][i])/2  elif (i==10):  split=(arr[0][i-1]+1)/2  """If the value is lower than the split, go to the left node,  if the value is larger, go to the right node."""  for j in range(10):  if(arr[0][j]<=split):  if(arr[1][j]==(-1)):  low\_0+=1  elif(arr[1][j]==1):  low\_1+=1  elif(arr[0][j]>split):  if(arr[1][j]==(-1)):  high\_0+=1  elif(arr[1][j]==1):  high\_1+=1  #Calculate and add the gini of each divided thing.  if((low\_0+low\_1)==0):  gini\_left=0  gini\_right=1-pow((high\_0/(high\_0+high\_1)),2)-pow((high\_1/(high\_0+high\_1)),2)    elif((high\_0+high\_1)==0):  gini\_left=1-pow((low\_0/(low\_0+low\_1)),2)-pow((low\_1/(low\_0+low\_1)),2)  gini\_right=0    else:  gini\_left=1-pow((low\_0/(low\_0+low\_1)),2)-pow((low\_1/(low\_0+low\_1)),2)  gini\_right=1-pow((high\_0/(high\_0+high\_1)),2)-pow((high\_1/(high\_0+high\_1)),2)    gini=(gini\_left+gini\_right)  #Store the largest gini and change the split\_result.  if (min\_gini>=gini):  min\_gini=gini  split\_result=split  if(low\_0<=low\_1):  val=1  else:  val=-1    return split\_result, val  bagging=np.zeros((10,2,10))  for i in range(10):  for j in range(10):  #bootstrap to randomly extract values  ran=random.randrange(10)  bagging[i][0][j]=df.loc[ran,'x']  bagging[i][1][j]=df.loc[ran,'y']  split, val=decision\_split\_value(bagging[i])  print("\n------------------------ Round {} ------------------------".format(i+1))  print(bagging[i])  print("Split value = ",split)  #Fill the table with splits and val values  for k in range(1,11):  x=k/10  if(x<=split):  result.loc[i,x]=val  else:  result.loc[i,x]=-val    #Find and fill each sum.  for i in range(1,11):  x=i/10  sum\_x = sum(result.loc[0:9,x])  result.loc[10,x] = sum\_x  #Find and fill each sign.  for i in range(1,11):  x=i/10  if(result.loc[10,x]>0):  result.loc[11,x] = 1  else:  result.loc[11,x] = -1  result=result.rename(index={0:1, 1:2, 2:3,3:4, 4:5, 5:6, 6:7, 7:8, 8:9, 9:10, 10:'Sum', 11:'Sign'})  print("\n------------------------- Result -------------------------\n",result) |

|  |
| --- |
| Screen shot |
|  |