Justin Cai Decision Tree

**Outlook**

**->Overcast**

**('Overcast', 'Yes')**

**->Sunny**

**-->Humidity**

**('High', 'No')**

**('Normal', 'Yes')**

**->Rain**

**-->Wind**

**('Strong', 'No')**

**('Weak', 'Yes')**

**[Finished in 0.1s]**

from \_\_future\_\_ import division

import math

import csv

class dtree():

def \_\_init\_\_(self, file):

reader = csv.reader(open(file, 'rb'))

self.size = len(list(reader)[0])-1

self.data = [[] for i in range(self.size-1)]

self.dic = [{} for i in range(self.size)]

self.cat = []

self.pos = [[] for i in range(self.size-1)]

self.play = []

self.create(file)

ent = []

for x in range(len(self.data)):

ent.append(self.entropy(self.data[x], self.play))

self.first = self.data[ent.index(min(ent))]

cate = self.cat[ent.index(min(ent))]

print(cate)

self.make\_tree(self.first, self.play, self.cat.index(cate))

#self.freqs = {a: self.data.count(a) for a in set(self.data)}

def make\_tree(self, curr, play, cate):

freqs = list(set(curr))

if self.entropy(curr, play) is 0:

val = ''

if len(freqs) is 1:

for x in self.dic[self.size-1]:

if self.dic[self.size-1][x] is play[0]:

val = x

print(curr[0], val)

else:

for f in freqs:

val = ''

for p in range(len(curr)):

if curr[p] == f:

for x in self.dic[self.size-1]:

if self.dic[self.size-1][x] is play[p]:

val = x

print(f, val)

return

if len(freqs) is 1:

ent = []

indices = [i for i, x in enumerate(self.data[cate]) if x == freqs[0]]

newlist = [[] for i in range(self.size-1)]

pos = []

for x in indices:

pos.append(self.play[x])

for c in range(self.size-1):

if c is not cate:

newlist[c].append(self.data[c][x])

for x in range(self.size-1):

if x is not cate:

ent.append(self.entropy(newlist[x], pos))

ind = ent.index(min(ent))

if ind >= cate and ind is not self.size-2:

ind += 1

print("-->" + self.cat[ind])

self.make\_tree(newlist[ind], pos, ind)

else:

ent = []

for f in freqs:

tl = []

tp = []

indices = [i for i, x in enumerate(curr) if x == f]

for x in indices:

tl.append(curr[x])

tp.append(play[x])

print("->"+f)

self.make\_tree(tl, tp, cate)

def entropy(self, DS, play):

freqs = {a: DS.count(a) for a in set(DS)}

trues = {a: 0 for a in set(DS)}

numpos = len(freqs)

for x in range(len(DS)):

if play[x] is 1:

trues[DS[x]]+= 1

total = len(DS)

entropy = 0

for i in trues:

prob = trues[i]/freqs[i]

if prob == 1 or prob ==0:

entropy += 0

else:

entropy += -prob\*freqs[i]/total\*math.log(prob,2)

return entropy

def create(self, file):

reader = csv.reader(open(file, 'rb'))

c = 0

count = [0] \* (self.size)

for row in reader:

if c is 0:

for x in range(1, self.size):

self.cat.append(row[x])

c+= 1

else:

for x in range(1, self.size):

if row[x] not in self.dic[x-1].values():

self.dic[x-1][count[x-1]] = row[x]

self.pos[x-1].append(count[x-1])

count[x-1] += 1

self.data[x-1].append(row[x])

if row[self.size] not in self.dic[self.size-1]:

self.dic[self.size-1][row[self.size]] = count[self.size-1]

count[self.size-1] += 1

# self.dic[self.size-1]['Yes'] = True

# self.dic[self.size-1]['No'] = False

self.play.append(self.dic[self.size-1][row[self.size]])

if \_\_name\_\_ == '\_\_main\_\_':

tree = dtree('tennis\_tree.csv')