QUIZ 1

1.Which is the right output? (B)

？ X is 55 + (6 / 2).

A. X = 55 + (6 / 2).

B. X = 58

C. Error

2.Which is the right output？ (A)

```
data(fact). data(rule).data(list).
cut_test_a(X) :-data(X).
cut_test_a('last clause').
cut_test_b(X) :-
  data(X),
  !.
cut_test_b('last clause').
```

？ -cut_test_b(X), write(X), nl, fail.

A.
```
fact
no
```

B
```
fact
rule
last clause
no
```

C.
```
fact
rule
list
last clause
no
```

D. **Error**

3. **Which is the right output？ (A)**

?-a(b,c,d) = a(X,X,d).

?-a(c,X,X) = a(Y,Y,b).

A. no no

B. no yes

C. yes no

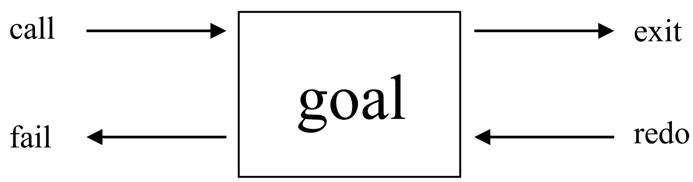
D. yes yes

4.**Terms / Data types in Prolog can be? (ABCD)**

A. Integer *B. Atom*

C. Variable *D. Structure*

1. Please write the correct ports in Prolog goal?



6. Predict the results of these unification queries.

?- a(b,c) = a(X,Y).
X = b,
Y = c

____.

?- a(X,c(d,X)) = a(2,c(d,Y)).
X = Y, Y = 2.

____.

?- a(X,Y) = a(b(c,Y),Z).
X = b(c, Z),
Y = Z.

____.

?- tree(left, root, Right) = tree(left, root, tree(a, b, tree(c, d, e))).
Right = tree(a, b, tree(c, d, e)).

____.

QUIZ 2

1. What are the four items necessary for a PEAS description?

Performance,
Environment,
Actuators,
Sensors

2. What is the *evaluation of a search strategy* while measuring performance of searching?

Completeness:
Can a solution eventually be found(Guaranteed)
Repeated states / loops / cycles can be avoided
Optimality:
Is the found solution the best among many solutions
Time complexity:
how long does it take to find a solution
Space complexity:
how much memory is needed to perform the search

3. Please fill in the white squares (LXX) in the table below, and the words to be selected are

L1	L2	L3	L4	L5	
L6		L7		L8	
L9	L10	L11	L12	L13	L14
L15				L16	

word(d,o,g).
word(r,u,n).
word(t,o,p).
word(f,i,v,e).
word(l,o,s,t).
word(m,e,s,s).
word(u,n,i,t).
word(f,o,r,u,m).
word(g,r,e,e,n).
word(s,u,p,e,r).
word(p,r,o,l,o,g).
word(v,a,n,i,s,h).
word(w,o,n,d,e,r).
word(y,e,l,l,o,w).

Try to write a rule solution.

solution(L1,L2,L3,L4,L5,L6,L7,L8,L9,L10,L11,L12,L13,L14,L15,L16):-
word(L1,L2,L3,L4,L5),
word(L9,L10,L11,L12,L13,L14),
word(L1,L6,L9,L15),
word(L3,L7,L11),
word(L5,L8,L13,L16).

QUIZ 3

1.Given a data set S, as it shows in the table below. Please calculate the *information gain* of 'outlook': Gain(S,Outlook)=?.

**Please give the necessary calculation steps.*

outlook	temperature	humidity	wind	playTennis
sunny	hot	high	weak	no
sunny	hot	high	strong	no
overcast	hot	high	weak	yes
rain	mild	high	weak	yes
rain	cool	normal	weak	yes
rain	cool	normal	strong	no
overcast	cool	normal	strong	yes
sunny	mild	high	weak	no
sunny	cool	normal	weak	yes
rain	mild	normal	weak	yes
sunny	mild	normal	strong	yes
overcast	mild	high	strong	yes
overcast	hot	normal	weak	yes
rain	mild	high	strong	no

$$Entropy(S) = -\ p_+ \log_2 p_+\ -\ p_- \log_2 p_-$$

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

1. Entropy(S) = -(9/14) log2 (9/14)
 -(5/14) log2 (5/14)
 = 0.94

2. Let 'Outlook' as the root node, we have:

Entropy(Sunny)= (-3/5)*log2(3/5)-(2/5)*log2(2/5)
=0.971 ;

Entropy(Overcast)= (-1)*log2(1)-(1)*log2(1)=0.0 ;

Entropy(Rain)=0.971

3. Gain(Outlook)=0.940-(5/14)*Entropy(Sunny)
-(4/14)*Entropy(Overcast)
-(5/14)*Entropy(Rain)
=0.247

More details please find:

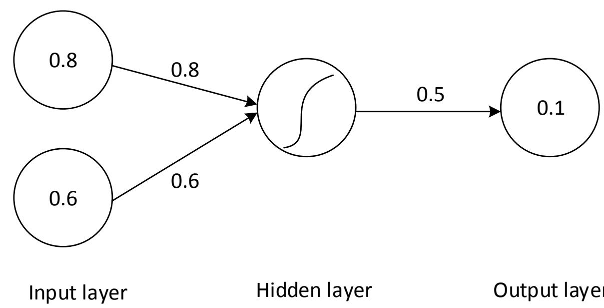
http://axon.cs.byu.edu/~martinez/classes/478/stuff/labhints/decisionTreeHelp.html

2. Given a table as below, please calculate the 1st and 2nd iteration for the weights using the data from the first row of the table according to the BP (Backpropagation) neural network.

Active function(sigmoid): f(x)= $\frac{1}{1+e^{-x}}$

Learning rate：0.3

X_1	X_2	Y
0.8	0.6	0.1
0.5	-0.1	0.7
⋮	⋮	⋮



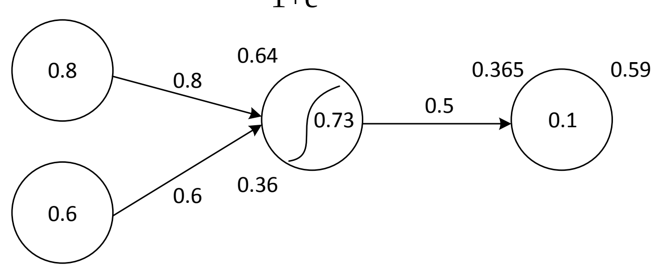
1. Calculate the value of hidden and output layer

O(hidden input)=0.8*0.8+0.6*0.6=1

O(hidden output)= $\frac{1}{1+e^{-1}}$ =0.73

O(output input)=0.73*0.5=0.365

O(output output)= $\frac{1}{1+e^{-0.365}}$ =0.59



2. Error calculation for the 1st iteration

$\delta_k = o_k(1-o_k)(t_k - o_k)$
=0.59*(1-0.59)*(0.1-0.59)=-0.1185

$\delta_h = o_h(1-o_h)\sum_{k \in outputs} w_{kh}\delta_k$
=0.73*(1-0.73)*0.5*(-0.12)= -0.01168

$\Delta w_o = \eta \delta_k x_h$ =0.3*(-0.1185)*0.73=-0.026

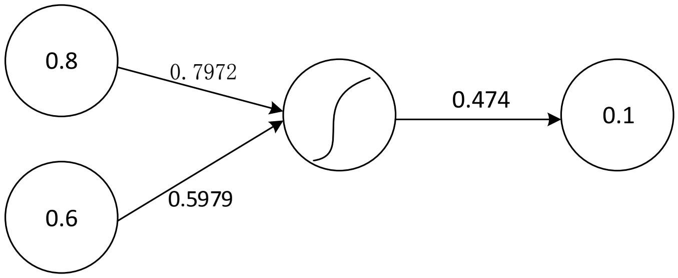
$\Delta w_{11} = \eta \delta_h x_{11}$ =0.3*(-0.01168)*0.8=-0.0028

$\Delta w_{21} = \eta \delta_h x_{21}$ =0.3*(-0.01168)*0.6=-0.0021

$w_o = w_o + \Delta w_o = 0.5 + (-0.026) = 0.474$

$w_{11} = w_{11} + \Delta w_{11} = 0.8 + (-0.0028) = 0.7972$

$w_{21} = w_{21} + \Delta w_{21} = 0.6 + (-0.0021) = 0.5979$



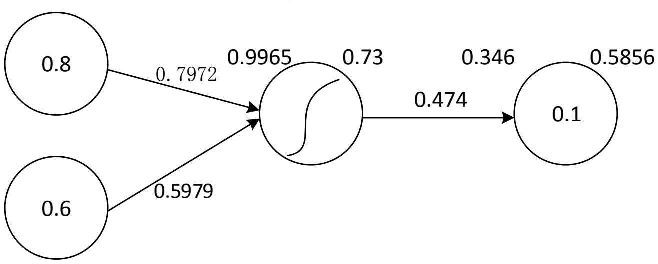
3. Calculate the value of hidden and output layer of 1st iteration

O(hidden input)=0.8* 0.7972+0.6*0.5979=0.9965

O(hidden output)= $\frac{1}{1+e^{-0.9965}}$ =0.73

O(output input)= 0.73*0.474=0.346

O(output output)= $\frac{1}{1+e^{-0.346}}$ = 0.5856



4. Error calculation for the 2st iteration

$\delta_k = o_k(1-o_k)(t_k - o_k)$

$=0.5856*(1-0.5856)*(0.1-0.5856)=-0.1178$

$\delta_h = o_h(1-o_h)\sum_{k\in outputs} w_{kh}\delta_k$

$=0.73*(1-0.73)*0.474*(-0.1178)= -0.0110$

$\Delta w_o = \eta\delta_k x_h = 0.3*(-0.1178)*0.73=-0.0258$

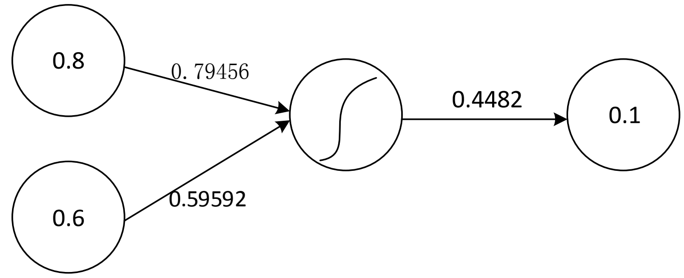
$\Delta w_{11} = \eta\delta_h x_{11} = 0.3*(-0.0110)*0.8=-0.00264$

$\Delta w_{21} = \eta\delta_h x_{21} = 0.3*(-0.0110)*0.6=-0.00198$

$w_o = w_o + \Delta w_o = 0.474 + (-0.0258) = 0.4482$

$w_{11} = w_{11} + \Delta w_{11} = 0.7972 + (-0.00264) = 0.79456$

$w_{21} = w_{21} + \Delta w_{21} = 0.5979 + (-0.00198) = 0.59592$



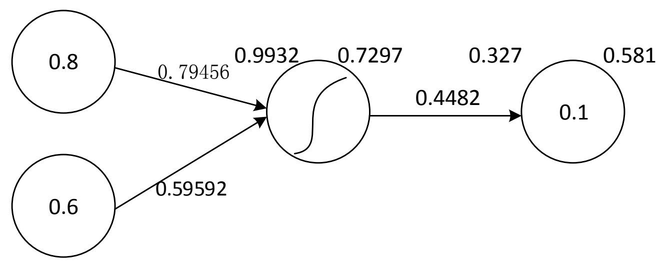
5. Calculate the value of hidden and output layer of 2nd iteration

O(hidden input)=0.8* 0.79456+0.6*0.59592
=0.9932

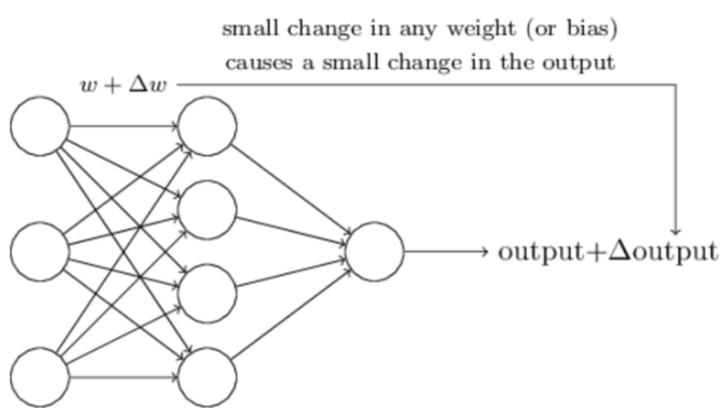
O(hidden output)= $\frac{1}{1+e^{-0.9932}}$ =0.7297

O(output input)= 0.7297*0.4482=0.327

O(output output)= $\frac{1}{1+e^{-0.327}}$ = 0.581



This one hidden layer network is very simple just for easy calculation by hand, following shows another sample of a normal BP neural network to solve the XOR problem, which has 3 input layer, 4 hidden layer and one output layer coded with Python.



BP_XOR.py:

```
# -*- coding: utf-8 -*-

import numpy as np

#Input data

X = np.array([[1,0,0],
            [1,0,1],
            [1,1,0],
            [1,1,1]])
#label
Y = np.array([[0,1,1,0]])
#Initial weights，[-1,1]
V = np.random.random((3,4))*2-1
W = np.random.random((4,1))*2-1
print(V)
print(W)
#learning rate
lr = 0.11

def sigmoid(x):
   return 1/(1+np.exp(-x))
#derivation
def dsigmoid(x):
   return x*(1-x)

def update():
   global X,Y,W,V,lr
#Input,output,weights,weights,learn rate
   # L1：Input -->hidden ；3X4
   # L2：hidden--> output；1
   L1 = sigmoid(np.dot(X,V))#output of hidden
layer(4,4)
   L2 = sigmoid(np.dot(L1,W))#output of output
layer(4,1)

   # L2_delta：error of output
   # L1_delta：error of hidden
   L2_delta = (Y.T - L2)*dsigmoid(L2)
   L1_delta = L2_delta.dot(W.T)*dsigmoid(L1)

   # W_C：weights changing from input to
hidden layer
   # V_C：weights changing from hidden to
output layer
   W_C = lr*L1.T.dot(L2_delta)
   V_C = lr*X.T.dot(L1_delta)

   W = W + W_C
   V = V + V_C

for i in range(20000):
   update()#updata weights
   if i%500==0:
      L1 = sigmoid(np.dot(X,V))#output of hidden
layer(4,4)
      L2 = sigmoid(np.dot(L1,W))#output of output
layer(4,1)
      print('Current Error:',np.mean(np.abs(Y.T-
L2)))

L1 = sigmoid(np.dot(X,V))#output of hidden
layer(4,4)
L2 = sigmoid(np.dot(L1,W))#output of output
layer(4,1)
print(L2)

def judge(x):
   if x>=0.5:
      return 1
   else:
      return 0

for i in map(judge,L2):
   print(i)

   #test a new data
X = np.array([[1,0,1],
            [1,1,1],
            [1,1,0],
            [1,0,0]])

L1 = sigmoid(np.dot(X,V))#output of hidden
layer(4,4)
L2 = sigmoid(np.dot(L1,W))#output of output
layer(4,1)

def judge(x):
   if x>=0.5:
      return 1
   else:
      return 0
#
for i in map(judge,L2):
 print(i)
```