1(a)

$$I(V)=\sum_{i=1}^{n} -P_i\log_2 P_i$$

$I(V)=0$ when $P_i = 0$

By convention, $0 \times \log_2 0 = 0$

In this case, information content of this classification is zero, which means that the selected variable V is helpless for the classification.

$$I(V) = \log_2 n$$

In this case, Variable V can be used to classified all the case since the information content of it is maximum, which is 1.

1(b)

Proof by contradiction: assume **Criminal(West)** is **false**
let *x* = ***West***, *y* = ***Mi***, *z* = ***Nono***

$$R1: \neg Am(x) \lor \neg We(y) \lor \neg Se(x,y,z) \lor \neg Ho(z) \lor Cr(x)$$
$$R2: \neg Mi(x) \lor \neg Ow(Nono,x) \lor Se(West,x,Nono)$$
$$R3: \neg Em(x,America) \lor Ho(x)$$
$$R4: \neg Mi(x) \lor We(x)$$
$$R5: Ow(Nono, M_i)$$
$$R6: Mi(M_i)$$
$$R7: Am(West)$$
$$R8: En(Nono, America)$$

and, Am() = America(), We() = Weapon(), Se() = Sell(), Ho() = Hostile(), Cr() = Criminal(),
Mi() = Missile(), Ow() = Own(), En() = Enemy()

$$\neg Am(West) \lor \neg We(M_i) \lor \neg Se(West,M_i,Nono) \lor \neg Ho(Nono) \lor Cr(West) \qquad \neg Cr(West)$$

$$Am(West) \qquad \neg Am(West) \lor \neg We(M_i) \lor \neg Se(West,M_i,Nono) \lor \neg Ho(Nono)$$

$$\neg We(M_i) \lor \neg Se(West,M_i,Nono) \lor \neg Ho(Nono) \qquad Mi(M_i)$$

$$En(Nono, America) \qquad \neg Se(West,M_i,Nono) \lor \neg Ho(Nono)$$

$$\neg Se(West,M_i,Nono) \qquad Ow(Nono, M_i) \qquad Mi(M_i)$$

$$(empty)$$

Therefore, **Criminal(West)** must be **true**

2(a)

$$O = f\left(\sum_{i=0}^{n} w_i \cdot I_i\right)$$

2(b)

$$h_{i,k} = f\left(\sum_{i} w_i \cdot N_{i,k}\right)$$

$$\text{E}=\frac{1}{2}\sum_{m=1}^{H_{K+1}} (O_m - T_m)^2 \quad , \qquad O_m = f\left(\sum_{i=1}^{H_{K+1}} w_i \cdot h_{i,k}\right)$$

$$O_m = f\left(\sum_{i=1}^{H_{K+1}} w_i \cdot f\left(\sum_{i} w_i \cdot N_{i,k}\right)\right)$$

2(c)

$$f(z) = \frac{1}{1+e^{-z}} = \left(1+e^{-z}\right)^{-1}$$

$$f\,'(z) = \left(1+e^{-z}\right)\left(1+e^{-z}\right)\left(e^{-z}\right)$$

$$f\,'(z) = f(z) \times \left(\frac{1+e^{-z}-1}{1+e^{-z}}\right)$$

$$f\,'(z) = f(z)\,(1 - f\,(z))$$

2(d)

Learning rate α can reduce the training time of the neural network. Since learning process in the neural network is slow, α is introduced to the equation. By adjusting it, for example, a bigger value at the beginning iterations, and then gradually decrease its value, it can speed up the training process and ensure the neural will not learn too fast which will cause a large inaccuracy.

2(e)

(i) f is the activation function

$$\frac{\partial}{\partial w_{K,j,k}}\mathrm{E} = \frac{\partial}{\partial O_k}\mathrm{E} \cdot \frac{\partial}{\partial w_{K,j,k}} O_k$$

$$\frac{\partial}{\partial w_{K,j,k}}\mathrm{E} = \frac{\partial}{\partial O_k}\left(\frac{1}{2}\sum_k (O_k\text{-}T_k)^2\right) \frac{\partial}{\partial w_{K,j,k}} O_k$$

$$\frac{\partial}{\partial w_{K,j,k}}\mathrm{E} = (O_k - T_k) \ \frac{\partial}{\partial w_{K,j,k}}\left(\sum_j h_{K,j} \cdot w_{K,i,k}\right)$$

$$\frac{\partial}{\partial w_{K,j,k}}\mathrm{E} = (O_k - T_k) f\,'(j)(h_{K,j}), where\ f(j) = \left(\sum_j h_{K,j} \cdot w_{K,i,k}\right)$$

(ii)

let k' be ^k

$$\frac{\partial}{\partial h_{i+1,k}}\mathrm{E} = \sum_{k'=1}^{H_{i+2}} \frac{\partial}{\partial\, h_{i+2,k'}}\mathrm{E} \cdot \frac{\partial}{\partial w i{+}1,k'}\ h_{i+2,k'}$$

$$\frac{\partial}{\partial h_{i+1,k}}E = \sum_{k'=1}^{H_{i+2}} \frac{\partial}{\partial\, h_{i+2,k'}}E \cdot (h_{i+2,k'})' \ \frac{\partial}{\partial w i{+}1,k'}\left(\sum_{j=1}^{H_{i+2}} w_{i+1,j,k'}, h_{i+1,j}\right)$$

$$\frac{\partial}{\partial h_{i+1,k}}E = \sum_{k'=1}^{H_{i+2}} \frac{\partial}{\partial\, h_{i+2,k'}}E \cdot h_{i+2,k'}\ (1 - h_{i+2,k'}) \cdot w_{i+1,j,k'}$$

$$\frac{\partial}{\partial h_{i+1,k}}\mathrm{E} = \sum_{k'=1}^{H_{i+2}} \varDelta_{i+2,\,k} \cdot w_{i+1,j,k'}\ , where\ \varDelta_{i+2,\,k} = \frac{\partial}{\partial\, h_{i+2,k'}}\mathrm{E} \cdot h_{i+2,k'}\ (1 - h_{i+2,k'})$$

(iii)

$$\frac{\partial}{\partial w_{i,j,k}} \mathrm{E} = \frac{\partial}{\partial h_{i+1,k}} \mathrm{E} \cdot \frac{\partial}{\partial w_{i,j,k}} h_{i+1,k}$$

$$\frac{\partial}{\partial w_{i,j,k}} \mathrm{E} = \left(\sum_{k'=1}^{H_{i+2}} \Delta_{i+2,\,k} \cdot w_{i+1,j,k'} \right) \cdot \frac{\partial}{\partial w_{i,j,k}} h_{i+1,k}$$

$$\frac{\partial}{\partial w_{i,j,k}} \mathrm{E} = \left(\sum_{k'=1}^{H_{i+2}} \Delta_{i+2,\,k} \cdot w_{i+1,j,k'} \right) \cdot (h_{i,k}(1-h_{i,k})) \frac{\partial}{\partial w_{i,j,k}} \left(\sum_{j=1}^{H_i} w_{i,j,k} \cdot h_{i,j} \right)$$

$$\frac{\partial}{\partial w_{i,j,k}} \mathrm{E} = \left(\sum_{k'=1}^{H_{i+2}} \Delta_{i+2,\,k} \cdot w_{i+1,j,k'} \right) \cdot (h_{i,k}(1-h_{i,k})) \cdot (h_{i,k})$$

(iv)

backward propagation algorithm:

Initialize the network weights (by assign random values to weight, for example)
do
    for each training example
        compute error of output layer
        compute e(i) in hidden layers
        compute e(ii) in output layers
        update all weights in network
until all examples are classified or met stopping criteria
return network