EVALUACIÓN #3

IDS
Usando
Red Neuronal Híbrida

Prof. NIBALDO RODRÍGUEZ A.

Implementación: Algoritmo Aprendizaje

Algoritmo de Aprendizaje

- **Inicializados: valores aleatorios**

Pesos Ocultos

$$r = \sqrt{\frac{6}{n_i + n_h}}$$

$$w^1, b^1 = rand\ (n_h, n_i) \times 2 \times r - r$$

Capa Oculta

$$h_j = f\left(\sum_{i=1}^{n_i} w^1_{j,i} \times x_i + b^1_j \right)$$

$$H = f(z) = \frac{1}{1 + \exp(-z)}$$

Pesos de Salida

$$w^2 = Y \times H^T \times \left(H \times H^T + \frac{I}{C} \right)^{-1}$$

Pseudo-Code
Phyton

train.py

```
Load train_input y train_label
Load param_config
# Data entrada  : Xe(d, N),      d: entrada, N: muestras
# Data salida     : Ye(nC, N),  nC: número de clases
# Nodos Ocultos: L,    Penalidad Pinv: C

[w1 bias w2]  = upd_pesos(Xe, Ye, L, C)
#costo=calc_costo(Xe,Ye,w1,bias,w2)
#Grabar pesos del IDS
save('pesos', w1,bias,w2)
#Grabar  vector de Costo
#savetxt( 'costo.csv', costo)
```

Function upd_date(xe,ye, L, C)

```
[Dim N]       = size(xe)
w1            = rand_W(L, Dim) ;          # Weigth hidden
bias          = rand_W(L,1)               # Bias  hidden
biasMatrix    = repmat(bias,1,N)          # bias Matrix

z             = w1*xe +biasMatrix
H             = Activation (z)

#-Calculate output weights
yh        = ye*H';
hh        = (H*H‘+eye(L)/C);
inv       = pinv(hh);
w2        = yh*inv;

return(w1, bias, w2)
```

Endfunction

rand_W.py

```
Function rand_W(next_nodes,current_nodes)

#W-random into [-r,r]

r=sqrt(6/(next_nodes + current_nodes))
w=rand(next_nodes,current_nodes)*2*r-r;

return(w)

Endfunction
```

test.py

```
Load test_input y test_label      # Data de testing
Load pesos                        # Pesos  estrenados
# Calcular data de salida del IDS
z=forward(Xv, w1,bias,w2)
# Calcular métrica de rendimiento
[accuracy, Fscore]=metrica(z, Yv)

#Grabar  Exactitud y r F-score de cada Clase
savetxt( 'fscore.csv', Accuracy, Fscore)
```

forward.py

```
Function [zv]=forward(xv, w1, bias, w2)

[D N]        = size(xv);
biasMatrix = repmat(bias,1,N)
z               = w1*xv +biasMatrix
H               = Activation(z);
z=w2*H;

return(z)

EndFunction
```

Lenguaje Implementación: PYTHON v. 3.6.7 window