CENTRO DE CIENCIAS BÁSICAS

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

INGENIERÍA EN COMPUTACIÓN INTELIGENTE

APRENDIZAJE INTELIGENTE

6° "A"

DR. FRANCISCO JAVIER LUNA ROSAS

Joel Alejandro Espinoza Sánchez

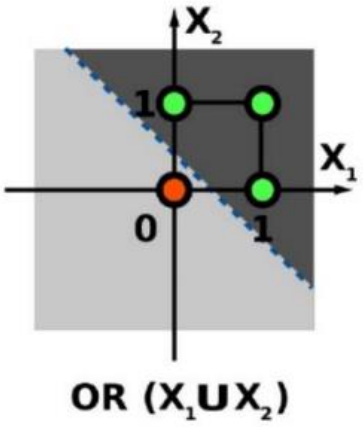
Dariana Gómez Garza

Fernando Francisco González Arenas

Fecha de entrega: Aguascalientes, Ags. 24 de febrero de 2021

PERCEPTRÓN

1. Diseñe una red neuronal de una capa (perceptrón) para la tabla de verdad OR



X1	X2	OR
0	0	0
0	1	1
1	0	1
1	1	1

Figura 1. Perceptrón para la tabla de verdad OR

Para obtener los pesos de w1, w2 y el umbral de teta se realizó una función específicamente para cada tabla de verdad OR, AND y AND negado. Las funciones llamadas `neuronaOR`, `neuronaAND`, `neuronaANDnegado` contiene los vectores que corresponden a `x1`, `x2` y el espacio `OR`, `AND` y el `AND` negado. Una vez realizados los vectores recurrimos a realizar un `for` que recorre desde la posición cero hasta la posición tres. Obtenemos el valor de `n` en la posición que se encuentre el ciclo y se hacen las comparaciones. Si n>= a `teta` la variable a valdrá uno de lo contrario, valdrá cero y si a es diferente de cero retornará un cero.

```
15
16  #*************************************neurona OR**********************
17
18  neuronaOR<-function(w1,w2,teta){
19
20
21    x1 <- c(0, 0, 1, 1)
22    x2 <- c(0, 1, 0, 1)
23    or <- c(0, 1, 1, 1)
24
25    tablaOR<- matrix(c(x1, x2, or),
26                     nrow = 4,
27                     ncol = 3)
28
29    colnames(tablaOR) <- c('x1', 'x2', 'OR')
30    tablaOR
31
32
```

```
33      for (x in 0:4)
34      {
35        n <-w1*x1[x] + w2*x2[x]
36        if(!is.null (n >= teta))
37        {
38          a <- 1
39        }else
40        {
41          a <- 0
42        }
43
44        if(!is.null (a!= or[x])) returnValue (0)
45
46      }
47      returnValue (1)
48
49    }
50
```

Una vez terminada la función tendremos un `do-while` en donde se da el valor de w1, w2 y teta aleatoriamente y con un condicional comparamos los valores obtenidos anteriormente en la función.

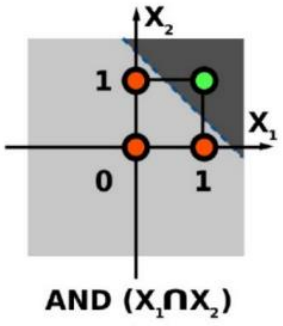
```
repeat{
  w1<-runif(1, min=-2, max=2)
  w2<-runif(1, min=-2, max=2)
  teta<-runif(1, min=-2, max=2)

  w1
  w2
  teta


  if(neuronaOR(w1,w2,teta)!=0){
    break
  }
}

#resultados neurona OR:
tablaOR
w1
w2
teta
```

2. Diseñe una red neuronal de una capa (perceptrón) para la tabla de verdad AND



X1	X2	AND
0	0	0
0	1	0
1	0	0
1	1	1

Figura 2. Perceptrón para la tabla de verdad AND

```r
#**********************************neurona AND**********************

neuronaAND <- function(w1,w2,teta){

  x1 <- c(0, 0, 1, 1)
  x2 <- c(0, 1, 0, 1)
  and <- c(0, 0, 0, 1)

  tablaAND<- matrix(c(x1, x2, and),
                    nrow = 4,
                    ncol = 3)

  colnames(tablaAND) <- c('x1', 'x2', 'AND')
  tablaAND


  for (x in 0:4)


  for (x in 0:4)
  {
    n <-w1*x1[x] + w2*x2[x]
    if(!is.null (n >= teta))
    {
      a <- 1
    }else
    {
      a <- 0
    }

    if(!is.null (a!= and[x])) returnValue (0)

  }
  returnValue (1)
}


 repeat{
   w1<-runif(1, min=-2, max=2)
   w2<-runif(1, min=-2, max=2)
   teta<-runif(1, min=-2, max=2)



   if(neuronaAND(w1,w2,teta)!=0){
     break
   }
 }


 #resultados neurona AND:
 tablaAND
 w1
 w2
 teta
```

3. Diseñe una red neuronal de una capa (perceptrón) para negar la tabla de verdad AND. Es decir, encuentre los pesos w1, w2 y el umbral TETA para negar la Red Neuronal que se muestra en el gráfico de la figura 2.

```
#**********************************neurona AND negado**********************

neuronaANDnegado<-function(w1,w2,teta){

  x1 <- c(0, 0, 1, 1)
  x2 <- c(0, 1, 0, 1)
  andnegado <- c(1, 1, 1, 0)


  tablaANDnegado<- matrix(c(x1, x2, andnegado),
                          nrow = 4,
                          ncol = 3)

  colnames(tablaANDnegado) <- c('x1', 'x2', 'ANDnegado')
  tablaANDnegado

  a <- 0
```

```
  a <- 0

  for (x in 0:4)
  {
    n <-w1*x1[x] + w2*x2[x]


    if(!is.null (n >= teta))
    {
      a <- 1
    }else
    {
      a <- 0
    }


    if(!is.null (a!= andnegado[x])) returnValue (0)
    break

  }
  returnValue (1)

}
```

```
w1<-0
w2<-0
teta<-0


repeat{
  w1<-runif(1, min=-2, max=2)
  w2<-runif(1, min=-2, max=2)
  teta<-runif(1, min=-2, max=2)



  if(neuronaANDnegado(w1,w2,teta)!=0){
    break
  }
}


#resultados neurona AND negado:
tablaANDnegado
w1
w2
teta
```

CONCLUSIÓN

Joel Alejandro Espinoza Sánchez:

Mediante el desarrollo de esta práctica, pudimos poner a prueba los aspectos estocásticos del funcionamiento en una red neuronal y observar la importancia de las ponderaciones dentro de este mismo algoritmo, así como la distribución de trabajo en equipo que pareció ser la óptima. Así se consiguió un gran trabajo y rendimiento entre los tres integrantes del equipo para aplicar nuestros conocimientos de las redes neuronales en el lenguaje R y creo que aprendimos el modelo básico de una red neuronal con un ejemplo muy básico como es el comportamiento de una compuerta lógica.

Dariana Gómez Garza:

Al realizar este trabajo pusimos en práctica como realizar una red neuronal (llamada perceptrón) en programación, ya que anteriormente habíamos realizado prácticas hechas a mano con ayuda del maestro.

Se realizó esta misma practica en C y además intentamos realizarla en R, que es de la cual se entrega este documento. Pudimos observar lo importante que es saber cómo realizar este tipo de trabajos para la IA y aunque fue algo muy sencillo, nos damos cuenta como se puede utilizar en algo tan simple como una compuerta lógica.

Fernando Francisco González Arenas:

Con la elaboración del perceptrón o neurona simple, tuvimos nuestro primer acercamiento con las redes neuronales, programando neuronas para procesar las tablas de verdad del OR y el AND lógico, dando nuestros primeros pasos para después resolver problemas mucho más complejos utilizando el aprendizaje de este tipo de inteligencia artificial. Con la realización de estas neuronas simples, aprendí a observar cómo trabajan las neuronas simuladas con programación, para que hagan cálculos y muestren como salida lo que deseamos.

REFERENCIAS

- Software for Data Analysis: Programming with R, Springer Verlag, N. Y., USA 2008. ISBN 978-0-387-75935-7
- Marsland Stephen. Machine Learning: An Algorithmic Perspective. Chapman & Hall/CRC 2009.