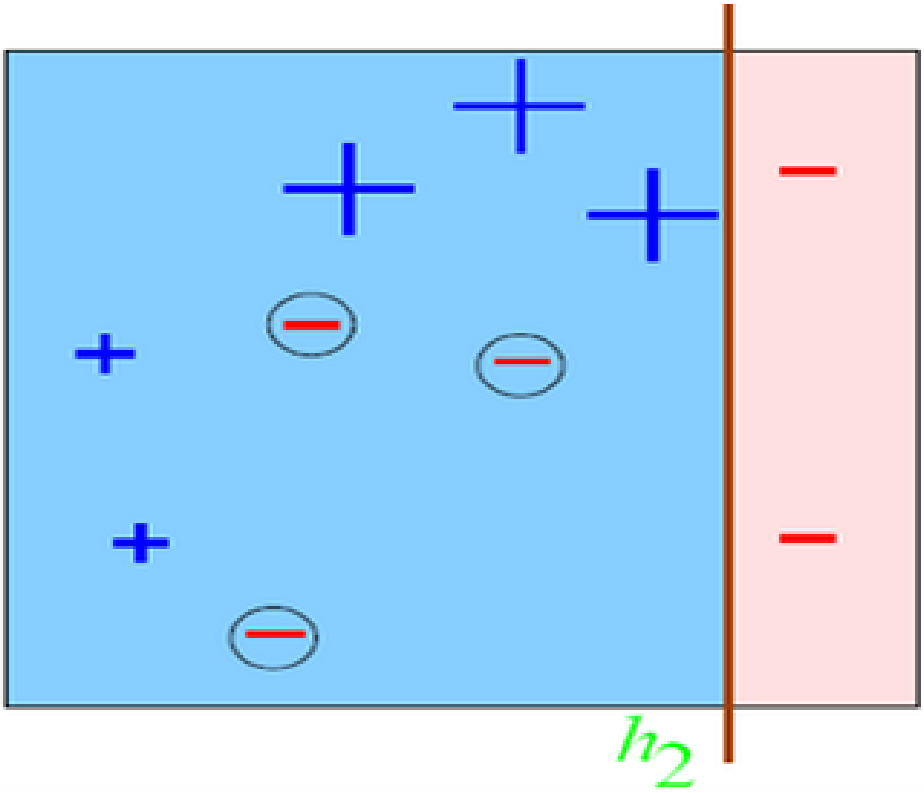
ADABOOST



Niamh Leia Eades Ellis

Grupo ARA

Nlee1@alu.ua.es

22-12-2019

Sistemas Inteligentes

Indice

Clasificador Débil

En primer lugar, hablaremos del clasificador débil. El clasificador débil es el objeto más pequeño que tenemos en Adaboost

```
public class ClasificadorDebil
{
    public int pixel;//dimension: 0 - 784
    public int umbral;//numero en la dimension 0 - 255
    public int direccion; //en qué lado está < o >
    public double error;//Suma de los que están mal
    public double confianza;//formula del log
    public static int h[];//valor asignado
    public static int dimension = 784;
    public static final int POSITIONS = 255;
    public static final int AMOUNTOFTRIES = 15;
```

En primer lugar, los atributos clave son el píxel (el número de dimensión que tenemos que se elige al azar un número entre 0 y 784, las dimensiones disponibles); el umbral (el número en la dimensión seleccionada que puede ser entre y 255) y en último lugar hacia qué dirección decimos que tenemos esa clase (hacia la izquierda o hacia la derecha de lo seleccionado).

Los otros atributos indicados son el error de cada clasificador débil y su confianza, el valor asignado por el clasificador, las dimensiones que existen, el número de umbral (posiciones) y el número de intentos que hace cada clasificador débil antes de elegir el final).

```
ClasificadorDebil()
{//hacer muchos, quedar con el pek
    pixel = (int) (Math.random() * dimension);
    umbral = (int) (Math.random() * POSITIONS);
    direccion = (int) (Math.random() * 2);

    //confianza = (1/2 * (Math.log((1 - error)/error) / Math.log(2)));
}
```

Como indicado anteriormente, pixel, umbral y dirección se eligen al azar.

```
public int ClasifyOne(ArrayList<Integer> x)
{
    int h = 0;
     if(x.get(pixel) < umbral)//si dir es menor, o sea, 1
            {
                if(direccion == 1)//no está ahí. H es -1
                   h = -1;
                else
                   h = 1;
            }
            else
            {
                 if(direccion == 1)
                    h = 1;
                else
                     h = -1;
            }
    return h;

}
```

En la función ClasifyOne, llenamos el array con -1 o 1, según el Clasificador débil.

Clasificador Fuerte

El clasificador fuerte es un conjunto de clasificadores débiles. Contiene el arrayList de todas las imágenes x, el array de las soluciones y, el nombre de la clase (en nuestro caso un número de cero a 9) y el conjunto de clasificadores débiles.

```
public class ClasificadorFuerte
{
    public ArrayList<ArrayList<Integer>> x;
    public ArrayList<Integer> y;
    public int name;
    public ArrayList<ClasificadorDebil> debiles = new ArrayList<ClasificadorDebil>();
```

En generarClasifAzar, generamos un clasificador débil aleatorio con la dimensión de los datos pasada por por parámetro.

```
public ClasificadorDebil generarClasifAzar(int dimension_de_los_datos)
{//tam objetos
    ClasificadorDebil.dimension = dimension_de_los_datos;
    ClasificadorDebil cd = new ClasificadorDebil();
    return cd;
}
```

En el constructor, además de asignar los valores, también creamos tantos constructores débiles como indicados. En este ejemplo se crean 10.

```
ClasificadorFuerte(int clase, ArrayList<ArrayList<Integer>> x, ArrayList<Integer> y)
{
    this.x = x;
    this.y = y;
    name = clase;
    for(int i = 0; i < 10; i++)
    {//nuevos clasificadores débiles
        debiles.add(generarClasifAzar(x.get(i).size()));
    }
}
```

Adaboost

Adaboost es un algoritmo importante de inteligencia artificial que aprende automáticamente a través de clasificadores. En nuestro adaboost tenemos lo siguiente:

```
public class Adaboost
{
    private int[][] X;
    private int [] y;
    private float[] d;//peso
    private ArrayList<ClasificadorFuerte> classes;
    private int numClases;
```

Por una parte, tenemos nuestro conjunto de imágenes y soluciones reales a cada uno de ellos (x e y), d, que es el peso que tiene cada imagen (varía según se equivoca el clasificador de solución), un clasificador fuerte por clase y la cantidad de clases.

```
Adaboost(Integer[] clases, ArrayList<ArrayList<Integer>> x, ArrayList<Integer> y) throws IOException
{// int[cantidad de imagenes][768] X
    d = new float[y.size()];
    int numClases = clases.length;
    int num;
    if(numClases > d.length)//Para solo hacer un bucle para tanto d como las clases
        num = d.length;
    else
        num = numClases;
    for (int i = 0; i < num; i ++)
    {
        if(i < numClases)
            classes.add( new ClasificadorFuerte(clases[i], x, y));
        if(i < d.length)
            d[i] = 1/d.length;
    }
    addToFile("Adaboost");
    //FICHERO
```

Main

```
for(int i = 0; i < 10; i++)
{//Cada clase

    ArrayList imgs = ml.getImageDatabaseForDigit(i);

    int trainer = (int) (imgs.size() * 0.8);
    for(int j = 0; j < imgs.size(); j++)
    { //Pasar imagen a int
        ArrayList  pixeles = new ArrayList<Integer>();
        Imagen img = (Imagen) imgs.get(j);
        for(int k = 0; k < img.getImageData().length; k++)
            pixeles.add(Byte2Unsigned(img.getImageData()[k]));

            if (j <= trainer)
            {
                train.add(pixeles);
                trainClass.add(i);
            }
            else
            {
                test.add(pixeles);
                testClass.add(i);
            }
    }
}
```

En nuestro main, lo más importante es conseguir las imágenes, convertirlas de Imagen a byte y de byte a integer y separarlas en dos grupos: las imágenes de la parte de train y las de la parte del test.

Después de esto, se llama a Adaboost y realiza todo lo anterior.