SISTEMAS INTELIGENTES

PRÁCTICA 2

Visión Artificial y Aprendizaje

Francisco Javier Pérez Martínez – Grupo 01
74384305M
UNIVERSIDAD DE ALICANTE | CURSO 2020/2021

Contenido

Introducción

En esta práctica se pretende desarrollar un sistema capaz de distinguir imágenes entre las diferentes categorías: aviones, automóviles, pájaros, gatos, ciervos, perros, ranas, caballos, barcos y camiones.

Para ello, se ha utilizado la base de datos de imágenes CIFAR-10 para el entrenamiento de AdaBoost, el algoritmo de aprendizaje automático empleado en nuestra práctica.

Este algoritmo, mediante un entrenamiento iterativo de los clasificadores débiles, les asigna mayor importancia a los datos mal clasificados anteriormente, y de esta manera obtiene un nuevo clasificador. De esta forma, logra adaptarse y obtener mejores resultados aumentando la precisión del algoritmo.

Objetivos

- Entender el funcionamiento del aprendizaje automático supervisado y en concreto el procedimiento de Boosting.
- Comprender el término de imagen y píxel, y cómo tenemos la posibilidad de aprender a diferenciar entre imágenes desde la información que surge en ellas.
- Comprender el papel de un clasificador débil en procedimientos de Boosting.
- Llevar a cabo el algoritmo AdaBoost así como un clasificador débil de umbral.
- Utilizar AdaBoost al problema de categorización de reconocimiento de objetos en fotografías.
- Hacer un estudio cuantitativo de la tasa de aciertos obtenida por medio de este procedimiento de aprendizaje.

AdaBoost

Explicación código implementado

En primer lugar, necesitaremos generar el conjunto de datos X e Y para poder trabajar con el algoritmo AdaBoost posteriormente, dónde X es el conjunto de imágenes para el entrenamiento e Y es la clasificación correcta de la imagen correspondiente.

```
public static void fillXandY(ArrayList<Imagen> X, int[] Y, ArrayList<Imagen> Xte
{
    CIFAR10Loader ml = new CIFAR10Loader();
    ml.loadDBFromPath("./cifar10_2000");
    if(choose){
        int indice = 0;
        for(int i=0;i<=9;i++){
            ArrayList dlimgs = ml.getImageDatabaseForDigit(i);
            if(i==tipoImagen){ // accedemos a la clase indicada i
                for(int j=0;j<(int) porcentaje;j++){
                    Imagen img = (Imagen) dlimgs.get(j);
                    X.add(img);
                    Y[indice] = 1;
                    indice++;
                }
            } else {
                // Resto de imagenes.
                for(int j=0;j<(int) porcentaje/9;j++){
                    Imagen img = (Imagen) dlimgs.get(j);
                    X.add(img);
                    Y[indice] = -1;
                    indice++;
                }
            }
        }
    } else {
```

En primer lugar, hay que destacar que utilizo una variable booleana para controlar en la función cuando ejecuta la parte de entrenamiento y cuando la de test. Cuando es true, ejecutaremos la parte de entrenamiento y cuando sea false la parte de test.

Para rellenar este conjunto de datos, he realizado un for de 0 a 9, es decir, hasta el número de categorías que tenemos. En la primera iteración cuando i==0 quiere decir que está en la carpeta de imágenes de aviones y por tanto rellenaré el conjunto de X e Y con un for hasta el porcentaje aplicado para el entrenamiento. En cuanto al resto de clases, las relleno hasta dicho porcentaje divido entre las 9 clases restantes.

Para rellenar el conjunto de test, siendo la booleana false, pero sabiendo que ahora empezamos el for desde dónde nos quedamos hasta el total de imágenes que tenemos en cada carpeta y para el resto de las imágenes el for se ejecutará hasta el total de imágenes menos el porcentaje divido entre las 9 clases restantes.

```
        } else {
            int indice = 0;
            for(int i=0;i<=9;i++){
                ArrayList dlimgs = ml.getImageDatabaseForDigit(i);
                if(i==tipoImagen){ // accedemos a la clase indicada i
                    for(int j=(int) porcentaje;j<total;j++){
                        Imagen img = (Imagen) dlimgs.get(j);
                        Xtest.add(img);
                        Ytest[indice] = 1;
                        indice++;
                    }
                } else {
                    // Resto de imagenes.
                    for(int j=0;j<(total-porcentaje)/9;j++){
                        Imagen img = (Imagen) dlimgs.get(j);
                        Xtest.add(img);
                        Ytest[indice] = -1;
                        indice++;
                    }
                }
            }
        }
    }
```

Entrenamiento

Una vez generado los conjuntos X e Y, he creado la clase clasificadorDebil.java en la cual almacenaré y utilizaré los siguientes atributos:

```
private int pixel; // 0..3071
private int umbral; // 0..255
private int direccion; // 1 o -1
private int [] pertenece;
private double error;
private double confianza;
```

- El **pixel** de la imagen, el cual vamos a utilizar para clasificar formado por 3072 dimensiones (32x32x3).
- El **umbral**, que nos dirá qué valor numérico nos va a servir para determinar si pertenece o no pertenece. Se debe encontrar entre 0 y 255.
- La **dirección**, que nos dirá si debemos comprobar si el umbral es mayor o menor.
- Un vector int [], para almacenar la comprobación entre el pixel y el umbral, tomando valores de +1 o -1 siguiendo unas condiciones que posteriormente mostraré.
- El **error**, para almacenar el error obtenido a partir de la suma de los pesos de aquellas imágenes que el clasificador no acierta.
- La **confianza**, nivel de confianza que tiene el clasificador débil en acertar la imagen, a menor error mayor confianza.

Una vez mencionado los atributos de la clase, mostraré los métodos usados para calcular y aplicar estos clasificadores débiles:

La siguiente función genera un clasificador débil al azar conteniendo el pixel, umbral y dirección.

```
public static clasificadorDebil generalClasifAzar(int size)
{
    Random random = new Random();

    int pixelRand =  (random.nextInt(3072));
    int umbralRand = (random.nextInt(256));
    int direccionRand = (random.nextInt(2));

    if(direccionRand == 0) direccionRand = -1;
    return new clasificadorDebil(pixelRand, umbralRand, direccionRand, size);
}
```

A continuación, se aplica un clasificador débil al conjunto de imágenes de entrenamiento. Aquí es dónde uso el vector int [] mencionado anteriormente para almacenar si pertenece o no a la clase aplicando una serie de condiciones mostradas a continuación:

```
public void aplicarClasifDebil(clasificadorDebil cD, ArrayList<Imagen> X)
{
    for(int i=0;i<X.size();i++){
        if(cD.direccion == 1){
            // Accedemos al imageData mediante el pixel
            if(X.get(i).getImageData()[cD.pixel] > cD.umbral){
                cD.pertenece[i] = 1;
            } else {
                cD.pertenece[i] = -1;
            }
        } else {
            if(X.get(i).getImageData()[cD.pixel] <= cD.umbral){
                cD.pertenece[i] = 1;
            } else {
                cD.pertenece[i] = -1;
            }
        }
    }
}
```

En la siguiente función, se obtiene el error sumando todos los pesos del vector D de aquellas imágenes que el clasificador no acierte:

```
public void obtenerErrorClasif(clasificadorDebil cD, ArrayList<Imagen> X, int[] Y, double[] D)
{
    double auxError = 0.0;
    for(int i=0;i<X.size();i++){
        if(Y[i] != cD.pertenece[i]){
            auxError += D[i];
        }
    }
    cD.error = auxError;
}
```

Todas estas funciones, las he usado en otra función auxiliar para el entrenamiento de un clasificador débil. Las cuáles se irán llamando hasta A (n.º de pruebas aleatorias). En esta función, devuelvo el clasificador débil que mejor clasifique, es decir, escogemos el que obtenga menor tasa de error.

```
public static clasificadorDebil trainClasifDebil(ArrayList<Imagen> X, int[] Y, double[] D)
{
    ArrayList<clasificadorDebil> clasDeb = new ArrayList<>();

    for(int i=1;i<=A;i++){
        clasificadorDebil cD = clasificadorDebil.generalClasifAzar(X.size());
        cD.aplicarClasifDebil(cD, X);
        cD.obtenerErrorClasif(cD, X, Y, D);
        clasDeb.add(cD);
    }
    clasificadorDebil mejor = null;
    mejor = clasDeb.get(0);
    for(int j=1;j<clasDeb.size();j++){
        if(mejor.getError() > clasDeb.get(j).getError()){
            mejor = clasDeb.get(j); //obtenemos el menor comprobando el que tenga menor tasa de error
        }
    }
    return mejor;
}
```

Una vez entrenado el clasificador y obtenido el que mejor clasifique, he utilizado la siguiente función la cual devuelve un clasificador fuerte:

```
public static clasificadorFuerte Adaboost(ArrayList<Imagen> X, int[] Y, ArrayList<Imagen> Xtest, int
{
    double[] D = new double[X.size()];
    ArrayList<clasificadorDebil> H = new ArrayList<>();

    for(int i=0;i<X.size();i++){
        D[i] = 1.0/(double) X.size(); // Indicamos como de dificil es de clasificar cada punto.
    }
    for(int t=1;t<=T;t++){
        // función que calcula y aplica clasificadores débiles
        // devolviendo el clasificador debil que mejor clasfica
        clasificadorDebil mejor = trainClasifDebil(X, Y, D);

        double confianza = 0.5 * Math.log((1.0-mejor.getError())/mejor.getError()) / Math.log(2);
        mejor.setConfianza(confianza);

        double Z = 0.0;
        double[] numD = new double [X.size()];
        //Actualizar pesos
        for(int i=0;i<X.size();i++){ //numerador
            double outClasif = -mejor.getConfianza() * Y[i] * mejor.getPertenece()[i];
            numD[i] = D[i] * Math.pow(Math.E, outClasif);
            Z += numD[i];// suma de todos los numeradores
        }
        for(int j=0;j<X.size();j++){
            D[j] = numD[j]/Z;
        }
        H.add(mejor);
    }
    return new clasificadorFuerte(H, X.size());
}
```

Está función, sería la equivalente al pseudocódigo de **AdaBoost** proporcionado en el enunciado de la práctica.

En primer lugar, tenemos nuestro vector de pesos D, el cual inicialmente estará inicializado a 1/X.size().

A continuación, tenemos un for hasta T (nº de clasificadores débiles a usar), dentro de este for llamaremos a nuestra función que devuelve el clasificador que mejor clasifica, calcularemos la confianza y actualizaremos el vector de pesos D siguiendo las fórmulas matemáticas del pseudocódigo. Finalmente, añadiremos este clasificador débil a un arraylist de clasificadores débiles y devolveremos un clasificador fuerte formado por sus clasificadores débiles.

Para devolver este clasificador fuerte, he creado la clase clasificadorFuerte.java la cual tiene los siguientes atributos:

```
private final ArrayList<clasificadorDebil> cDebiles;
private int [] pertenece;
```

- Un arraylist de clasificadores débiles, ya que un clasificador fuerte está formado por un conjunto de clasificadores débiles.
- Y, un vector de tipo int [] al igual que en la clase anterior para almacenar si el clasificador fuerte pertenece o no a una clase.

En esta clase, he utilizado la función para aplicar el clasificador débil como base, pero adaptada al clasificador fuerte. Al igual que la función de aplicar débil, esta sigue las mismas condiciones.

Almacenamos el resultado en H de multiplicar la confianza de cada clasificador débil * lo que dice cada clasificador débil que es la imagen.

En caso de H (hipótesis fuerte) mayor a 0, nuestro vector int [] será igual a 1 y en caso contrario igual a -1. En caso de ser igual a 1, el clasificador fuerte supone que la imagen es del tipo de clase, por ejemplo, avión. Y en caso contrario, que no es de ese tipo.

```
public void aplicarClasificadorFuerte(ArrayList<Imagen> X) {

    for(int i=0;i<X.size();i++){
        double H = 0.0;
        for(int j=0;j<this.cDebiles.size();j++){
            if(cDebiles.get(j).getDireccion() == 1){
                if(Adaboost.Byte2Unsigned(X.get(i).getImageData()[cDebiles.get(j).getPixel()]) > cDebiles.get(j).getUmbral()) {
                    H += cDebiles.get(j).getConfianza();
                }
                else {
                    H -= cDebiles.get(j).getConfianza();
                }
            } else {
                if(Adaboost.Byte2Unsigned(X.get(i).getImageData()[cDebiles.get(j).getPixel()]) < cDebiles.get(j).getUmbral()) {
                    H += cDebiles.get(j).getConfianza();
                }
                else {
                    H -= cDebiles.get(j).getConfianza();
                }
            }
        }
        if(H > 0) {
            pertenece[i] = 1;
        }
        else {
            pertenece[i] = -1;
        }
    }
}
```

Por último, en la siguiente función devuelvo un arraylist de clasificadores fuertes, es decir, los 10 clasificadores fuertes generados de cada clase.

```
public static ArrayList<clasificadorFuerte> AdaboostParaTodasLasClases() {

    int aciertosTotales = 0, aciertosTotalesTest = 0;
    int totalImagenes = 0, totalImagenesTest = 0;
    ArrayList<clasificadorFuerte> clasifFuertes = new ArrayList<>();

    System.out.println("\n" + "Estadísticas - Conjunto de Entrenamiento" + "\t\t" + "Estadísticas - Conjunto de Test" + "\n
    for(int i=0;i<=9;i++){
        File directorio = new File("./cifar10_2000/" + i);
        int total = totalArchivos(directorio);

        ArrayList<Imagen> X = new ArrayList<>();
        ArrayList<Imagen> Xtest = new ArrayList<>();

        double porcentaje = total * P;
        int[] Y = new int[(int) porcentaje * 2]; // Le asigno un tamaño mayor ya que sino daba error de ArrayIndexOutOfBoun
        int[] Ytest = new int[(int) porcentaje * 2];
        tipoImagen = i;
        fillXandY(X, Y, Xtest, Ytest, porcentaje, total, tipoImagen, true);
        fillXandY(X, Y, Xtest, Ytest, porcentaje, total, tipoImagen, false);
        clasificadorFuerte cF = Adaboost(X,Y,Xtest,Ytest);
        cF.aplicarClasificadorFuerte(X); // Aplicamos el clasificador fuerte para el conjunto de entrenamiento.

        int aciertos = X.size() - cF.obtenerErrorClasif(X, Y);
        aciertosTotales += aciertos;
        totalImagenes += X.size();

        cF.aplicarClasificadorFuerte(Xtest); // Aplicamos el clasificador fuerte para el conjunto de test.
        int aciertosTest = Xtest.size() - cF.obtenerErrorClasif(Xtest, Ytest);
        aciertosTotalesTest += aciertosTest;
        totalImagenesTest += Xtest.size();

        // Mostrar estadísticas
        mostrarAciertosDeCadaClase(i, X.size(), aciertos, Xtest.size(), aciertosTest);
        clasifFuertes.add(cF);
    }
    mostrarMediaAciertos(totalImagenes, aciertosTotales, totalImagenesTest, aciertosTotalesTest);
    return clasifFuertes;
}
```

Como se puede observar, aquí es dónde escribo el porcentaje para entrenamiento y para test, en mi caso 80% y 20% respectivamente (He elegido este porcentaje ya que en clase de prácticas se comentó que un buen porcentaje de entrenamiento rondaba entre 70-80% y para test un 20-30%.

Una vez dado estos porcentajes, llamo a la función que genera los conjuntos de X e Y tanto de entrenamiento como test, la función AdaBoost y la que aplica un clasificador fuerte.

Por último, imprimo las estadísticas de cada clase (% de acierto del clasificador fuerte) tanto para el conjunto de entrenamiento como para el de test para realizar próximamente un estudio de los resultados obtenidos.

Finalmente, en el main llamo a la función anterior guardándome el clasificador fuerte obtenido de cada clase y a la función que almacena estos clasificadores en un fichero para realizar posteriormente la lectura de éstos para el modo de test.

```
//Se ejecuta la práctica como entrenamiento
if (args[0].equals("-t")) {

    ArrayList<clasificadorFuerte> cFuertes = new ArrayList<>();

    cFuertes = AdaboostParaTodasLasClases();
    almacenarClasificadoresFuertes(cFuertes);
```

Test

Para la parte de Test, una vez almacenado los clasificadores fuertes en un fichero, pasaremos a leer y cargar el contenido de este fichero cuando se ejecute el proyecto en modo test.

```
} else {
    //Se ejecuta la práctica como test
    File ImageFile = new File("./" + args[1]); // le pasamos la imagen
    ArrayList<clasificadorFuerte> cFuertesFichero = new ArrayList<>();
    cFuertesFichero = leerFichero(); // leemos los 10 clasificadores fuertes
    Imagen image = new Imagen(ImageFile); // creamos la imagen con el archivo pasado
    ArrayList<Imagen> imagenes = new ArrayList<>();

    imagenes.add(image);

    int tipoClase = -1;
    double valor = 0.0;

    for(int i=0;i<cFuertesFichero.size();i++){
        clasificadorFuerte cF = cFuertesFichero.get(i);
        double auxH = cF.obtenerHipotesisFinal(imagenes); //obtenemos los valores de H
        System.out.println("auxH = " + auxH);

        if(valor < auxH){ //comprobamos que H es la mayor
           valor = auxH;
           tipoClase = i;
        }
    }
    // Imprimimos la H mayor con su respectiva representación del objeto (imagen).
    System.out.println("\nClasificación de la imagen \n");
    System.out.println("-----------------------------------------");
    System.out.println("La imagen " + args[1] + " representa al objeto: " + tipoClase);
```

En la ejecución de la parte de Test, leemos los clasificadores fuertes de nuestro fichero y clasificamos la imagen pasada como parámetro.

En la siguiente captura, se muestra que el archivo *"imagen.png"* representa al **objeto 8**.

```
auxH = 2.817444762062995
auxH = 0.7657831146345542
auxH = 1.0277571310514668
auxH = 2.514458234739261
auxH = -3.3184270143476113
auxH = -1.3761176571682565
auxH = -5.70917562382852
auxH = 1.357116845358562
auxH = 3.851910687422198
auxH = -5.319097951138603
```

Para ello, he comprobado cual de todas las Hipótesis finales es la mayor y, por tanto, decir que la imagen representa a dicha clase.

En este caso particular, la prueba acierta y esta imagen si pertenece a la clase. Pero, si volviéramos a ejecutar la prueba, con otro conjunto de clasificadores fuertes, puede ser que acierte o puede que no, dándose incluso el caso de contemplar confusión entre 2 clases si con los mismos clasificadores probamos a intercambiar la imagen.

```
Clasificación de la imagen

-----------------------------------------
La imagen imagen.png representa al objeto: 8
BUILD SUCCESSFUL (total time: 0 seconds)
```

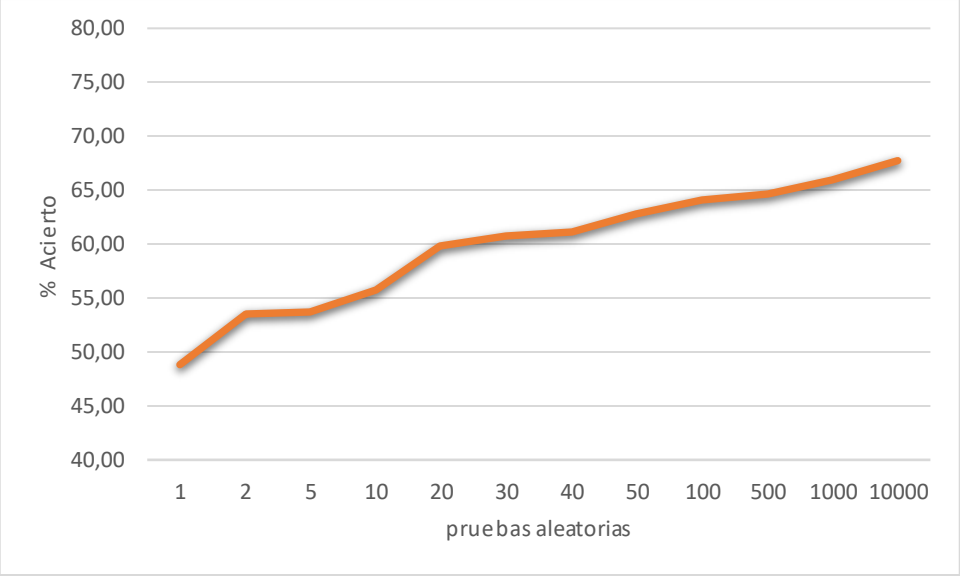
Cuestiones propuestas

- **¿Cuál es el número de clasificadores que se han de generar para que un clasificador débil funcione? Muestra una gráfica que permita verificar lo que comentas.**

En primer lugar, para que un clasificador débil funcione, hay que suponer que éste debe tener un porcentaje de acierto de las imágenes mayor o igual al 50%.

Para ello, mediante la siguiente tabla utilizando los 10 clasificadores fuertes de cada clase con 1 clasificador débil cada uno comprobaremos que numero de clasificadores débiles debemos generar para obtener un porcentaje adecuado cambiando el n. º de pruebas aleatorias:

Nº A	1	2	5	10	20	30	40	50	100	500	1000	10000	100000
1	49,36	66,56	50,32	54,46	56,05	61,20	63,69	66,88	66,56	69,43	67,52	69,43	70,06
2	51,35	50,00	48,31	54,05	65,20	65,88	60,81	61,49	67,91	65,88	67,91	69,93	70,27
3	53,40	52,47	52,47	54,01	55,56	56,79	61,42	58,95	58,95	60,80	60,49	64,2	63,89
4	49,51	55,02	54,05	55,34	56,31	57,28	55,66	57,28	60,84	60,52	61,81	63,11	63,75
5	35,38	56,73	55,56	59,36	57,31	57,60	61,40	61,70	64,62	65,79	65,20	68,13	68,42
6	49,66	50,34	56,90	55,86	62,76	61,38	63,45	64,48	62,76	65,86	66,90	67,93	68,28
7	45,87	53,52	55,66	59,94	64,53	69,11	62,39	65,44	67,58	66,97	69,42	70,03	72,78
8	58,33	50,00	55,77	53,53	54,49	54,65	58,09	61,54	61,22	61,78	62,18	66,03	66,35
9	45,37	49,69	58,02	56,79	65,12	64,20	62,04	67,59	67,28	62,22	68,83	69,14	72,53
10	50,00	50,31	50,00	53,70	60,49	60,04	61,42	62,35	62,65	67,59	68,52	69,44	70,68
Media	**48,82**	**53,46**	**53,71**	**55,70**	**59,78**	**60,81**	**61,04**	**62,77**	**64,04**	**64,68**	**65,88**	**67,74**	**68,70**



Como vemos en la gráfica anterior, el % medio de acierto de los clasificadores débiles aumenta a medida que aumentamos el n. º de pruebas aleatorias. Sin embargo, al ir aumentando este número de pruebas hace que el tiempo de ejecución de la práctica sea excesivo por lo que no he podido mostrar por ejemplo cuando tengamos 1Millon de pruebas aleatorias teniendo en cuenta que solo he generado 1 débil en cada clasificador fuerte. En caso de tener más débiles, bastaría con un número menor de pruebas para que el tiempo de ejecución aumente.

En definitiva, a partir de más de un 55%, cuando A está entre 10 y 20, empezamos a ver que los clasificadores débiles funcionan.

- **¿Cómo afecta el número de clasificadores generados al tiempo empleado para el proceso de aprendizaje? ¿Qué importancia le darías? Justifica tu respuesta**.

Cuanto más clasificadores generemos más tiempo emplearemos, obviamente, pero llegará un punto en el que es innecesario añadir más puesto que ya no va a conseguir más porcentaje de acierto debido al sobre entrenamiento.

- **¿Cómo has dividido los datos en conjunto de entrenamiento y test? ¿Para qué es útil hacer esta división?**

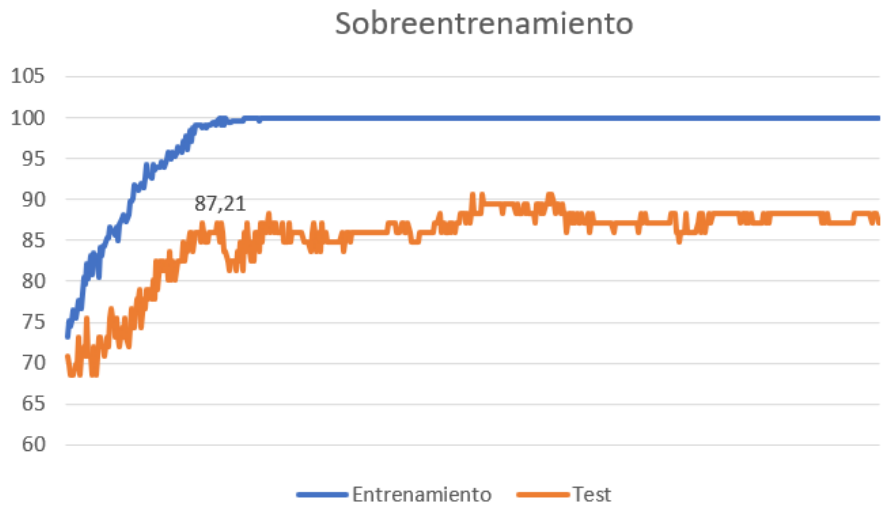
Como he explicado anteriormente, he usado un 80% en entrenamiento y un 20% en test, ya que en prácticas se comentó que un buen porcentaje rondaba entre 80-70% y 20-30% respectivamente. Esto es útil de esta forma ya que para comprobar que esté funcionando correctamente el reconocimiento de las imágenes usaremos distintas imágenes ya que si usáramos las mismas con las que entrenamos no estaríamos obteniendo resultados reales.

- **¿Has observado si se produce sobre entrenamiento? Justifica tu respuesta con una gráfica en la que se compare el error de entrenamiento y el de test a lo largo de las ejecuciones.**

NOTA: En mi caso, he comparado tasa de acierto de entrenamiento y de test ya que lo he implementado en el código de esta forma, pero básicamente el resultado sería la inversa de la gráfica de error. Con valores de T= 500 y A = 100

Sí, pero llega un punto que entreno demasiado el clasificador fuerte y empieza a fallar para el conjunto de test. A partir del 87% por mucho que sigamos generando clasificadores débiles va a tender a devolver el mismo resultado para test. Por este razonamiento, deducimos que es innecesario seguir generando más.

También, llega un punto que en entrenamiento alcanza un 100% por lo que no está aprendiendo sino memorizando ya que el algoritmo empieza a quedarse ajustado a unas características muy concretas de los datos de entrenamiento sin relación al objetivo deseado.



- **Comenta detalladamente el funcionamiento de AdaBoost teniendo en cuenta que tasa media de fallos obtienes para aprendizaje y test.**

El funcionamiento del algoritmo se ha ido explicando anteriormente durante la explicación del propio código implementado.

En mi caso, he mostrado por pantalla el porcentaje de acierto del clasificador fuerte de cada clase y la media de estos porcentajes. Esta ejecución toma valores de T=25 y A=10.

```
run:

Estadísticas - Conjunto de Entrenamiento              Estadísticas - Conjunto de Test

----------------------------------------              ----------------------------------------
Clase: 0                                              Clase: 0
El numero de aciertos sobre 314 imágenes es: 224      El numero de aciertos sobre 86 imágenes es: 64
% De acierto del clasificador fuerte: 71,34%          % De acierto del clasificador fuerte: 74,42%

----------------------------------------              ----------------------------------------
Clase: 1                                              Clase: 1
El numero de aciertos sobre 296 imágenes es: 226      El numero de aciertos sobre 84 imágenes es: 62
% De acierto del clasificador fuerte: 76,35%          % De acierto del clasificador fuerte: 73,81%

----------------------------------------              ----------------------------------------
Clase: 2                                              Clase: 2
El numero de aciertos sobre 324 imágenes es: 207      El numero de aciertos sobre 86 imágenes es: 50
% De acierto del clasificador fuerte: 63,89%          % De acierto del clasificador fuerte: 58,14%

----------------------------------------              ----------------------------------------
Clase: 3                                              Clase: 3
El numero de aciertos sobre 309 imágenes es: 220      El numero de aciertos sobre 84 imágenes es: 50
% De acierto del clasificador fuerte: 71,20%          % De acierto del clasificador fuerte: 59,52%

----------------------------------------              ----------------------------------------
Clase: 4                                              Clase: 4
El numero de aciertos sobre 342 imágenes es: 244      El numero de aciertos sobre 88 imágenes es: 65
% De acierto del clasificador fuerte: 71,35%          % De acierto del clasificador fuerte: 73,86%

----------------------------------------              ----------------------------------------
Clase: 5                                              Clase: 5
El numero de aciertos sobre 290 imágenes es: 206      El numero de aciertos sobre 82 imágenes es: 52
% De acierto del clasificador fuerte: 71,03%          % De acierto del clasificador fuerte: 63,41%

----------------------------------------              ----------------------------------------
Clase: 6                                              Clase: 6
El numero de aciertos sobre 327 imágenes es: 231      El numero de aciertos sobre 87 imágenes es: 58
% De acierto del clasificador fuerte: 70,64%          % De acierto del clasificador fuerte: 66,67%

----------------------------------------              ----------------------------------------
Clase: 7                                              Clase: 7
El numero de aciertos sobre 312 imágenes es: 216      El numero de aciertos sobre 85 imágenes es: 59
% De acierto del clasificador fuerte: 69,23%          % De acierto del clasificador fuerte: 69,41%

----------------------------------------              ----------------------------------------
Clase: 8                                              Clase: 8
El numero de aciertos sobre 324 imágenes es: 239      El numero de aciertos sobre 86 imágenes es: 64
% De acierto del clasificador fuerte: 73,77%          % De acierto del clasificador fuerte: 74,42%

----------------------------------------              ----------------------------------------
Clase: 9                                              Clase: 9
El numero de aciertos sobre 324 imágenes es: 241      El numero de aciertos sobre 86 imágenes es: 59
% De acierto del clasificador fuerte: 74,38%          % De acierto del clasificador fuerte: 68,60%

##########################################################################################################

Estadísticas Media - Conjunto de Entrenamiento        Estadísticas Media - Conjunto de Test

----------------------------------------              ----------------------------------------
El numero de aciertos sobre 3162 imágenes es: 2254    El numero de aciertos sobre 854 imágenes es: 583
% De acierto del clasificador fuerte: 71,28%          % De acierto del clasificador fuerte: 68,27%

BUILD SUCCESSFUL (total time: 15 seconds)
```

- **¿Cómo has conseguido que Adaboost clasifique entre los 10 dígitos cuando solo tiene una salida binaria?**

Ejecutando el algoritmo de AdaBoost 10 veces y a la hora de clasificar una imagen pasada por argumento, en caso de que varias digan que pertenece a una clase determinada miraremos cual de todas las Hipótesis finales es la mayor.

Finalmente, concluiré dejando el mejor porcentaje de acierto de entrenamiento y test que he encontrado, que toma los valores de T = 100 y A = 85, obteniendo de media un 95,7% de entrenamiento y un 83% de test.

```
run:

Estadísticas - Conjunto de Entrenamiento              Estadísticas - Conjunto de Test

----------------------------------------              ----------------------------------------
Clase: 0                                              Clase: 0
El numero de aciertos sobre 314 imágenes es: 298      El numero de aciertos sobre 86 imágenes es: 72
% De acierto del clasificador fuerte: 94,90%          % De acierto del clasificador fuerte: 83,72%

----------------------------------------              ----------------------------------------
Clase: 1                                              Clase: 1
El numero de aciertos sobre 296 imágenes es: 288      El numero de aciertos sobre 84 imágenes es: 70
% De acierto del clasificador fuerte: 97,30%          % De acierto del clasificador fuerte: 83,33%

----------------------------------------              ----------------------------------------
Clase: 2                                              Clase: 2
El numero de aciertos sobre 324 imágenes es: 309      El numero de aciertos sobre 86 imágenes es: 68
% De acierto del clasificador fuerte: 95,37%          % De acierto del clasificador fuerte: 79,07%

----------------------------------------              ----------------------------------------
Clase: 3                                              Clase: 3
El numero de aciertos sobre 309 imágenes es: 293      El numero de aciertos sobre 84 imágenes es: 67
% De acierto del clasificador fuerte: 94,82%          % De acierto del clasificador fuerte: 79,76%

----------------------------------------              ----------------------------------------
Clase: 4                                              Clase: 4
El numero de aciertos sobre 342 imágenes es: 315      El numero de aciertos sobre 88 imágenes es: 76
% De acierto del clasificador fuerte: 92,11%          % De acierto del clasificador fuerte: 86,36%

----------------------------------------              ----------------------------------------
Clase: 5                                              Clase: 5
El numero de aciertos sobre 290 imágenes es: 277      El numero de aciertos sobre 82 imágenes es: 66
% De acierto del clasificador fuerte: 95,52%          % De acierto del clasificador fuerte: 80,49%

----------------------------------------              ----------------------------------------
Clase: 6                                              Clase: 6
El numero de aciertos sobre 327 imágenes es: 317      El numero de aciertos sobre 87 imágenes es: 77
% De acierto del clasificador fuerte: 96,94%          % De acierto del clasificador fuerte: 88,51%

----------------------------------------              ----------------------------------------
Clase: 7                                              Clase: 7
El numero de aciertos sobre 312 imágenes es: 299      El numero de aciertos sobre 85 imágenes es: 68
% De acierto del clasificador fuerte: 95,83%          % De acierto del clasificador fuerte: 80,00%

----------------------------------------              ----------------------------------------
Clase: 8                                              Clase: 8
El numero de aciertos sobre 324 imágenes es: 317      El numero de aciertos sobre 86 imágenes es: 74
% De acierto del clasificador fuerte: 97,84%          % De acierto del clasificador fuerte: 86,05%

----------------------------------------              ----------------------------------------
Clase: 9                                              Clase: 9
El numero de aciertos sobre 324 imágenes es: 313      El numero de aciertos sobre 86 imágenes es: 71
% De acierto del clasificador fuerte: 96,60%          % De acierto del clasificador fuerte: 82,56%

##########################################################################################################

Estadísticas Media - Conjunto de Entrenamiento        Estadísticas Media - Conjunto de Test

----------------------------------------              ----------------------------------------
El numero de aciertos sobre 3162 imágenes es: 3026    El numero de aciertos sobre 854 imágenes es: 709
% De acierto del clasificador fuerte: 95,70%          % De acierto del clasificador fuerte: 83,02%

BUILD SUCCESSFUL (total time: 15 seconds)
```