Escuela Superior de Ingeniería

APRENDIZAJE COMPUTACIONAL

BackPropagation

AUTORES:
Juan Ruiz Bonald
Rafael Roman Aguilar

Puerto Real, 10/04/2019

1 Introducción teórica

Debido al amplio abanico de campos que tratamos en esta ocasión (todas las anteriores prácticas), podríamos basar esta introducción teórica en ellas; pero como hemos empleado un nuevo método para encontrar una solución a un problema mediante redes neuronales, lo basaremos en él, es por ello que vamos a tratar la introducción sobre **Backpropagation**.

Antes de nada, es importante resaltar la existencia de las redes neuronales **multicapas** con funciones de activación **NO lineales**, ya que jugarán un partido muy importante a la hora de utilizar el **Backpropagation**

Con suficientes neuronas ocultas se puede aproximar cualquier función, por esto se dice que las Redes Neuronales Multicapas con funciones de activación NO lineales están hechas para un propósito general, son flexibles y NO lineales.

para utilizar el **Backpropagation** necesitamos las Redes Neuronales anteriormente mencionada: multicapas y que posean una función de activación **NO lineal**. Esta última condición se debe a que la red debe ser derivable en todo su dominio.

La idea central detrás de este algoritmo es que los errores de cada neurona de las capas ocultas son determinados hacia atrás desde la capa de salida. Puede considerarse como una generalización de la regla delta para funciones no lineales y redes multicapas.

Pasando a hablar de las fases de este algoritmo:

1. Fase de Propagación

Un patrón de entrada es aplicado a la primera capa de neuronas en la red. Después propagamos hasta obtener una salida y comparamos esta con el resultado deseado, así de esta manera vemos el error obtenido por cada neurona.

2. Fase de Adaptación

Distribuye el error de un elemento de salida a todos los elementos ocultos a los que está conectado. Esta fase involucra un paso hacia atrás a través de la red donde la señal de error es transferido a cada elemento y variando el peso de las conexiones (de está manera es como la red va aprendiendo).

Por tanto, definimos el error como el error cuadrático medio y el objetivo es minimizar este. Este algoritmo está basado en el método del gradiente descendiente.

Para terminar, vamos a enumerar algunas de las principales características y problemas del **Backpropgation**.

Características:

- El algoritmo permite buscar el mínimo error de una función de un conjunto de patrones de entrenamiento.
- Requiere derivar la función de activación.
- Requiere cambiar los pesos de la red.

Problemas:
- Puede converger a un mínimo local
- Si la inclinación en el espacio de entrenamiento es pequeña, el entrenamiento es lento.

2 Pregunta y Respuesta

Mi pregunta planteada en el foro fue la siguiente:

Pregunta: Marque la afirmación sobre Backpropagation que es falsa:

1. Puede converger a un mínimo local.

2. Si se reduce la pendiente, el entrenamiento es muy lento.

3. Solo pueden utilizarse 2 capas ocultas.

4. Si la pendiente es cero, el algoritmo se detiene.

La respuesta correcta es la **tercera**. Puesto que el algoritmo Backpropagation acepta redes neuronales con más de 2 capas ocultas ya que esto depende del tipo de región de decisión.

3 Metodología

Primero, tratando de preprocesar los datos tuvimos como opción usar Fisher o PCA (para poder ver con claridad *explained*, la variable que nos dice la importancia de cada característica).

Empezaremos por explicar el método seguido ante la base **Wine**:
Observando de primera mano Fisher con 2 características podemos ver que los datos quedan bien diferenciados y ordenados. Cosa que no ocurre con PCA con el mismo número de características.
Es por eso, debido a los buenos resultados arrojados por Fisher, que decidimos usar Fisher con dos características en la base de datos Wine.
Cabe añadir que no normalizamos porque Fisher ya normaliza cuando lo ejecutamos.

Continuando con la base **Bank**:
Antes de nada, debido a que tenemos 41188 muestras, reduciremos aleatoriamente el numero de estas a 1000, ya que de las 41188 hay muchas que empeoran y estropean la muestra, debido a que son datos aislados.
Probando tanto Fisher como PCA observamos unos resultados pobres, y es por ello que decidimos aplicar PCA para ver las características más importantes (en este caso las dos primeras nos dan más de un 95%) y después normalizaremos

estos datos con las 2 características más importantes.

Por tanto, nos quedamos con 2 características en cada conjunto de datos.

A continuación veremos las estrategias seguidas en Backpropagation y en validación.

- En el algoritmo de backpropagation hemos escogido como función de activación una función sigmoide.
- Para poder entrenar los pesos de un conjunto de patrones hemos adaptado el algoritmo para que los acepte y opere con ellos aplicando matrices y producto de matrices.
- Una vez calculados los inputs de todos los patrones, para calcular el error de las capas ocultas y de la capa de salida, aplicamos las formulas que aparecen en la página 2 del documento de *"backpropagation.pdf"* del campus virtual.
- Ya con los errores de cada patron para modificar los pesos hacemos la media de todos los errores y aplicamos la formula para modificar los pesos que encontramos en el mismo sitio que en el punto anterior.
- Para poder actualizar los errores de las capas ocultas necesitamos los pesos sin modificar de la capa de salida por lo que utilizamos una variable w_old para almacenar los pesos antes de modificarlos.
- Para la validación hemos optado por utilizar el cross-validation con k = 10 debido a que es el metodo de validación más estandarizado con el que podremos comprarar mejor nuestro problema y como vimos en la práctica 2 es un buen metodo para validar.

4 Resultados Obtenidos

Los resultados obtenidos en esta práctica con nuestra implementacion del algoritmo backpropagation han salido con una tasa de acierto del 0% ya que nuestra implementación no devolvia de forma correcta los resultados esperados y por la falta de tiempo hemos tenido que utilizar el algoritmo visto en clase de teoría.

Con el algoritmo visto en teoria nos salen unos resultados muy buenos con unos errores muy bajos. Con el cross validation con k = 10 y un número de repeticiones de 30 nos salen unos buenos resultados con unos errores minimos.

Aquí podemos ver las gráficas de los errores del algoritmo del backpropagation para las bases de datos Wine y Bank

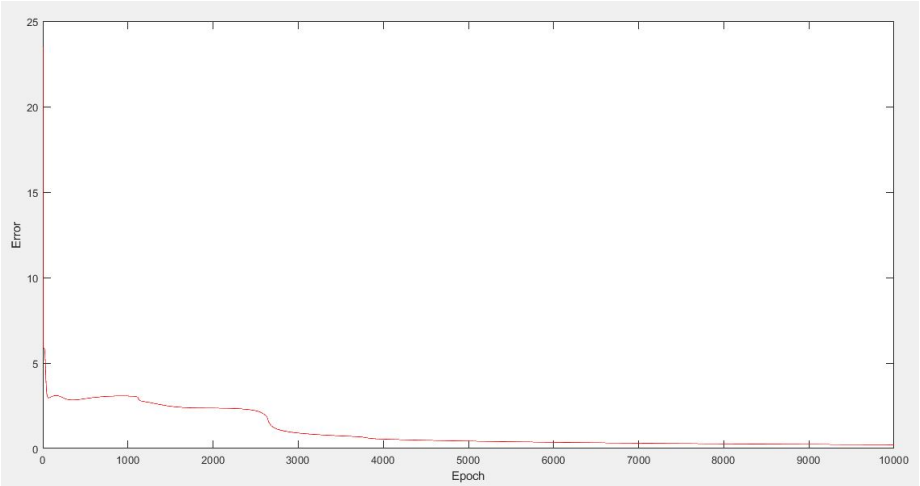

Figure 1: Error respecto a epoch de la base de datos Wine

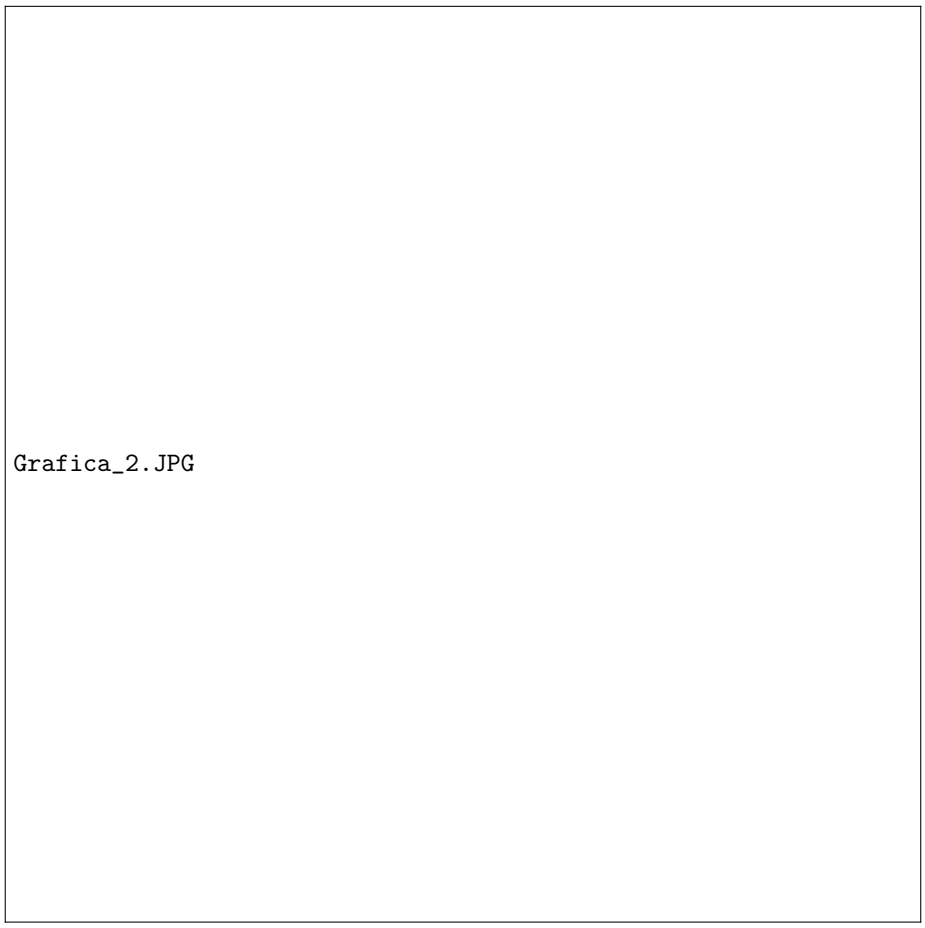

Figure 2: Error respecto a epoch de la base de datos Bank

5 Conclusiones

En esta práctica la coonclusión que he sacado ha sido que el backpropagation es un buen algoritmo para las redes neuronales y que estas sirven para muchos propositos diferentes.