# Conceptos acerca de redes neuronales

## 1. Minimización de la función de coste asociada.

Cuando se entrena una red neuronal, el objetivo principal es obtener el menor error posible a la hora de realizar una predicción de clases. Esta predicción de clases viene dada por una distribución de probabilidad:

$$P(x_1, x_2 \dots x_n|\sigma) = \prod_{i=1}^{n} P(x_i|\sigma)$$

El objetivo principal es obtener un vector:

$$(\sigma_1, \sigma_2 \dots \sigma_n)$$

Este vector va a estar formado por los conocidos **estimadores de verosimilitud**, que se corresponderán con los pesos de las conexiones, y que tiene como objetivo maximizar la distribución probabilidad que rige la **función de verosimilitud** (*likelihood function*). Maximizar esta función de verosimilitud va a implicar forzosamente que se minimice la **función de coste asociada** (si aumentamos *f(x)*, disminuimos *-f(x)*).

Es importante decir también que estas distribuciones de probabilidad vienen dadas por una función de separación de clases, ya sea lineal o no lineal. En este caso supondremos que la función de separación es una recta con ecuación:

$$y = wx + b$$

Para maximizar (o minimizar) una función, simplemente hay que estimar su derivada e igualarla a cero (de esta forma se obtienen los máximos y mínimos)

### 1.1 Datos que siguen una distribución continua (Gaussiana)

En este caso estudiaremos el caso en que los datos siguen una distribución continua (no suele ser así). Cuando es así, podemos expresar la función de separación de clases de la siguiente forma:

$$y \approx N(wx, b)$$

donde:

$$\mu = wx$$
$$\sigma = b(cte)$$

De esta forma la distribución de probabilidad queda de la siguiente forma:

$$P(x_i|\sigma) = P(y|wx)$$

Como hemos mencionado antes, el objetivo es minimizar el error de nuestra red o lo que es lo mismo, maximizar la función de verosimilitud de la distribución que siguen los datos. Por tanto debemos estimar la derivada de nuestra distribución de probabilidad:

$$\sigma = 1$$
$$\prod_{i=1}^{n} P(x_i|\sigma) = \sum_{i=1}^{n} log(P(x_i|\mu_i, \sigma)) =$$
$$\sum_{i=1}^{n} log\frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{1}{2\sigma}(y_i - \mu_i)^2)$$

Hemos supuesto el valor cte 1 para que al derivar el cuadrado se anule con *1/2* y de esta forma la expresión final sea más sencilla. Podemos ver que la exponencial y el logaritmo se anulan. Nuestra expresión final a derivar es la siguiente:

$$\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2}(y_i - \mu_i)^2$$

A continuación derivamos esta expresión:

$$\frac{\partial}{\partial \mu} \sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2}(x_i - \mu)^2 = \sum_{i=1}^{n}(y_i - \mu_i)$$

Por tanto la igualdad que buscábamos es:

$$\sum_{i=1}^{n}(y_i - \mu_i) = 0$$
$$\mu_i = w_i x_i$$
$$\sum_{i=1}^{n}(y_i - \mu_i) = \sum_{i=1}^{n}(y_i - w_i x_i)$$
$$\sum_{i=1}^{n}(y_i - w_i x_i) = 0$$

## 1.2 Datos que siguen una distribución discreta (Bernoulli)

Otra posibilidad (más frecuente) es que los datos sigan una distribución discreta. En este caso asumiremos que la distribución discreta que siguen es una Bernoulli.

En este caso la función de probabilidad cambia, puesto que la Bernoulli sigue la siguiente función de probabilidad.

$$P(y = k)\mu^{x_i}(1 - \mu)^{1 - x_i}$$

En este caso la distribución acepta solo dos valores (0,1). Nuestro objetivo sin embargo es el mismo que el anterior. Estimar la derivada de la función de verosimilitud para maximizarla.

$$\prod_{i=1}^{n} \mu^{x_i}(1 - \mu)^{1 - x_i} = \sum_{i=1}^{n} x_i log(\mu) + \sum_{i=1}^{n}(1 - x_i) log(1 - \mu) =$$
$$\sum_{x_i}^{n} x_i \frac{1}{log(\mu)} - \sum_{x_i}^{n}(1 - x_n)\frac{1}{log(1 - \mu)} = 0$$

Como antes, tenemos una función de separación de clases, en este caso lineal, con la misma forma que antes. Así, tenemos:

$$\mu = wx_i$$
$$\sum_{x_i}^{n} x_i \frac{1}{log(wx_i)} - \sum_{x_i}^{n}(1 - x_n)\frac{1}{log(1 - wx_i)} = 0$$

Finalmente obtenemos la función de verosimilitud, que es la siguiente:

$$\sum_{i=1}^{n} y_i log(w_i x_i) + (1 - y_i) log(1 - w_i x_i)$$

Podemos observar que esta función de verosimilitud obtenida es muy parecida a la función de **entropía cruzada** (básicamente es igual pero cambiada de signo)  La función de entropía cruzada (o cross entropy) es otro tipo de función de coste (antes obteníamos el error cuadrático medio, ahora hemos obtenido la función de entropía cruzada).

El objetivo es maximizar esa función de verosimilitud, y minimizar su inversa (recordamos, maximizar *f(x)* y minimizar *-f(x)*). Por tanto, vamos a minimizar la función de entropía cruzada, que para unos datos que siguen una distribución Bernoulli, es la función de coste asociada.

La distribución de Bernoulli está estréchamente relacionada al concepto de entropía en informática. La entropía en información mide la cantidad de incertidumbre. En este caso, medirá la incertidumbre de la predicción realizada por nuestra red neuronal. Para entender la relación, basta con saber que cuanto más cerca se esté de los extremos (0,1) más baja será la entropía (siendo 0 el mínimo), ya que la probabilidad de que sea 0 o 1 será máxima o mínima, pero no habrá dudas. Sin embargo, para valores en torno al 0.5, la entropía será máxima (1) ya que será igual de probable que el valor pertenezca a una clase que a otra y por tanto la incertidumbre es mayor.

## 1.3 Problema... ¿y si tenemos varias clases?

Si en nuestras entradas de datos, los datos se clasifican en varias clases, las dos alternativas anteriores no son válidas.  En este caso, podremos asumir (debemos asumir) que nuestros datos están regidos por una **distribución categórica**

A la hora de asumir que los datos siguen una distribución categórica, asumimos a su vez que las clases están codificadas en vectores de tamaño *nClases*. Poniendo como ejemplo que los datos se clasifican en 3 clases:

- [1,0,0] hace referencia a la clase 1
- [0,1,0] hace referencia a la clase 2
- [0,0,1] hace referencia a la clase 3

La función de probabilidad de la distribución categórica es la siguiente:

$$P(y) = \prod_{k=0}^{k-1} \pi_k^{y_k}$$

Donde *y* hace referencia al valor de la clase *k*.

**Ejemplo**

$$\pi = (0.4, 0.5, 0.1)$$

Este vector nos muestra las probabilidades de cada clase (en total suman 1 como podemos observar). Aplicando la función de probabilidad de la distribución categórica:

$$P(y = 0) = 0.4^1 * 0.5^0 * 0.1^0 = 0.4.$$

A continuación estimaremos la función de coste asociada a la distribución categórica:

$$\prod_{n=0}^{n-1} P(y_n|x_n) = \prod_{i=1}^{n} \prod_{j=1}^{m} P(y = k) = \prod_{i=1}^{n} \prod_{j=1}^{m} \pi_{ij}^{y_{ij}}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m} log(\pi_{ij}^{y_{ij}}) = \sum_{i=1}^{n}\sum_{j=1}^{m} y_{nk} log(\pi_{ij})$$

Es ciertamente parecida a la *cross entropy* (componente logarítmica). De hecho esta función de coste se llama *cross entropy categorical* (entropía cruzada categórica).

## 1.4 Conclusiones

En esta sección se ha visto que el objetivo principal será el de maximizar la función de verosimilitud y minimizar la función de coste asociada. Esto se lleva a cabo con la derivada de la función de probabilidad de la distribución que siguen los datos e igualando esta a 0 (obtenemos así máximos y mínimos).

En función del número de clases en que se clasifiquen nuestros datos, elegiremos unas distribuciones u otras:

- Si nuestros datos se clasifican solo en **dos clases (0,1)** (poco común): Hemos visto dos posibilidades: los datos pueden estar regidos por una distribución de probabilidad continua (en este caso hemos supuesto que Gaussiana) o por una distribución de probabilidad discreta (Bernoulli).
    - Para la Gaussiana hemos obtenido el siguiente gradiente de la función de verosimilitud, correspondiente con la inversa de la función de coste del error cuadrático medio (que sería con signo contrario)

$$\sum_{i=1}^{n}(y_i - w_i x_i)$$

    - Para la Bernoulli, al cambiar la función de probabilidad, también cambiará la función de verosimilitud y por tanto su derivada. En este caso, obtenemos la correspondiente a la inversa de la función de coste de la entropía curzada (cross entropy) o función de regresión logística

$$\sum_{i=1}^{n} y_i log(w_i x_i) + (1 - y_i) log(1 - w_i x_i)$$

- En cambio, lo más frecuente, si nuestros datos se clasifican en **varias clases**, las opciones previamente vistas no son válidas. Se introduce aquí como alternativa la distribución categórica. De la misma forma, presenta una función de verosimilitud que se tratará de maximizar. Para esta distribución la función de máxima verosimilitud es la siguiente:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} y_{nk} log(\pi_{ij})$$

## 2. Funciones de activación

En función de la distribución que sigan nuestros datos, va a ser necesario que nuestra red neuronal aplique una función de activación u otra. Generalmente, estas funciones debe ser continuas*, diferenciables y monótonamente no-decrecientes. Como hemos visto antes, el uso de las distribuciones depende de un criterio, el nº de clases que se usan para clasificar datos. Empleando el mismo criterio, se presentan como ejemplos las siguientes funciones de activación (las más comunes para cada caso):

- Para datos que se clasifican en dos clases (binaria por ejemplo), lo normal es utilizar una función de activación como la **sigmoide**. Esta función recibe un valor en el intervalo de los reales y a su salida lo mapea en el intervalo [0,1].
- Cuando los datos se clasifican en varias clases en cambio, el uso de la sigmoide es inviable, pero surgen alternativas como la la función **softmax**, de idéntico comportamiento que la sigmoide pero aceptando multitud de clases.

Sin embargo, está probado que la función que mejores resultados proporciona, especialmente en redes convolucionales o redes neuronales profundas en general es la función **ReLu**, función rampa o función rectificadora. Esta función, a diferencia de las mencionadas anteriormente, es lineal (sigmoide y softmax no lo son) y no devuelve valores acotados en un intervalo tan pequeño. Además, elimina los valores negativos (estableciéndolos a 0) y en cambio si da importancia a los positivos, de tal forma que la entrada de un valor negativo se transforma a  0 y no se propaga y la entrada de un valor positivo si será propagada por la red neuronal. *Su único inconveniente es que esta función no es continua en 0, por lo que hay que asegurarse que todas las entradas netas no sean nulas.