03 - Función SOFTMAX - REDES NEURONALES - (Parte 3) - Transcript

En este vídeo continuamos con nuestro modelo de red neuronal artificial de una sola capa.

Es importante notar que esta capa también funciona como capa de salida, por lo tanto es importante tener una función de activación que nos permita darle un mejor significado o interpretación a los resultados que obtenemos en cada una de las neuronas de esta capa de salida.

Por esto en este vídeo vamos a presentar la función de activación softmax, la cual es la función de activación más comúnmente utilizada en problemas de clasificación multiclase.

Ya que convierte cada uno de estos valores resultantes de una transformación lineal realizada en estas neuronas, a probabilidades que nos indican que tan probable es que determinada imagen pertenezca a una clase en particular.

Continuamos.

El modelo presentado en el vídeo previo es el de un clasificador lineal, donde recordemos que un clasificador es un programa computacional o un modelo computacional que recibe una entrada y en su salida determina a qué tipo de objeto pertenece dicha entrada.

En otras palabras determina a qué clase pertenece dicha entrada, de ahí el nombre de clasificador.

Este modelo fue representado por una red neuronal muy sencilla de una sola capa y cada neurona representa cada una de las clases que este modelo podría determinar.

Es decir, la primera neurona que tiene la salida marcada en verde representa la salida para la clase gato, la salida marcada en naranja representa la salida para la clase perro y finalmente la última representa la salida para la clase ave o pájaro.

También mencionamos que cada neurona tiene asociada una serie de pesos sinápticos los cuales van a afectar cada una de las entradas y en este caso las entradas son los píxeles de la imagen.

Por simplificación dijimos que solamente esta imagen tiene cuatro píxeles y cada píxel está conectado a cada una de las neuronas.

Mencionamos también que podemos representar estos pesos sinápticos en forma matricial donde cada fila representa los pesos asociados a una neurona.

Por ejemplo, esta fila representada en color verde son los pesos que están asociados a la primera neurona, la neurona del gato.

De tal forma que aquí tendríamos que esto sería 0.17, 0.32, etcétera.

De igual forma, los segundos, la segunda fila con los pesos en fondo color naranja son los de la segunda neurona y así los de la tercera fila son los de la tercera neurona.

Al multiplicarse por los píxeles de la imagen de entrada obtenemos un vector que al sumarle este vallas nos da este resultado.

Y este resultado son los valores de score o vamos a decirle puntuaciones que nuestro modelo está arrojando para cada clase.

Dos cosas importantes que notar acá.

Los pesos sinápticos asociados con cada neurona están identificados en esta matriz de pesos sinápticos que vamos a llamar W por weights en inglés.

Es la letra que usualmente se utiliza para indicar estas variables.

Y este valor de los biases o también conocidos como sesgos son valores que son sumados a cada neurona.

Hay un valor asociado a cada neurona y es un valor que te permiten decir qué tan grande debe ser esta suma resultado de haber multiplicado las entradas por los pesos sinápticos.

Qué tan grande debe ser este valor para resultar en una activación que sea significativa.

Entonces tenemos un vector columna de salida que indica los puntajes que están asociados con cada una de las clases.

Idealmente, dado que la imagen de entrada es la de un gato, quisiéramos que el puntaje más alto sea el de la clase asociada al gato.

Y en este caso podemos ver que efectivamente el puntaje más alto 3.32 corresponde al de la clase gato, mientras que el de perros 3.01 y 3.17 para el de la clase ave.

En este caso nuestra red neuronal sencilla está clasificando bien la imagen del gato.

De forma más generalizada también podemos representar todas estas imágenes, sus vectores de píxeles, los podemos agrupar también en una sola matriz haciendo un bloque o un batch.

A esto le vamos a llamar minibatch y después vamos a ver qué significa esto, que va a tener más relevancia en la implementación de modelos de redes neuronales más grandes.

Pero bueno, por lo pronto vamos a decir que este es un bloque de tres imágenes, un batch de tres imágenes que vamos a llamar x ahora mayúscula porque ya se trata de una matriz y podemos procesar así estos bloques de imágenes de entrada prácticamente en un solo paso al realizar una multiplicación de la matriz w por la matriz x y vemos que las dimensiones coinciden ya que recordemos que para que en una multiplicación de matrices, en este caso una matriz de tres por cuatro, al multiplicarse por otra matriz que sea de cuatro por tres, vamos a ponerle esta matriz es de cuatro por tres, de cuatro filas por tres columnas, nos va a resultar en una matriz que va a ser de tres filas por tres columnas.

Recordemos que las dimensiones interiores son las que deben de coincidir.

Al hacer esta operación y luego sumarle este bias lo que va a hacer es que se le va a sumar a cada columna, es una operación de tipo broadcast que nos permite sumarle un vector a cada vector de una matriz.

Posteriormente vamos a hablar de esto más a detalle.

El punto clave aquí es que el resultado nos da la predicción para cada una de las imágenes de entrada.

Por ejemplo, la imagen de entrada tipo gato nos dicen que tiene un valor, un puntaje de 3.32 gato, 3.01 perro y 3.17 ave.

Vamos a ver esta imagen más, está estos resultados más a detalle.

Aquí vemos de nueva cuenta la matriz de W de pesos que está siendo multiplicada por nuestra matriz de imágenes de entrada.

Se le suma el bias y nos dan estos resultados o puntajes para cada una de las imágenes de entrada que fue el gato, el perro y el ave.

Lo importante es darnos cuenta de algunos detalles de estos resultados.

Ya sabemos entonces que existe un valor asociado de pesos sinápticos y un valor asociado de bias o sesgos los cuales resultan para las imágenes de entrada en estos valores.

Ahora, ¿cómo encontramos una forma de medir qué tan buenos son los resultados que estamos obteniendo con estos valores?

Recordemos que podemos modelar nuestra red normal.

Vamos a decir que es una función llamada neural network NNM que recibe una imagen de entrada, datos de entrada, pesos W y biases.

Entonces con base en estas en estas entradas, la red normal es una función de todos estos elementos.

Sin embargo, tenemos que darnos cuenta que los únicos elementos que nosotros podemos controlar son los elementos W y los elementos de biases, los pesos y los sesgos.

No podemos cambiar la entrada de entrada.

Finalmente es lo que queremos que nuestra red neuronal aprenda a reconocer.

Volviendo a este punto, tenemos que las salidas de nuestro modelo de red neuronal arrojan un valor de 3.32 cuando vemos procesamos la imagen de un gato de forma que efectivamente el valor más alto corresponde al del gato y podemos decir que está funcionando bien los valores que tenemos actuales de W y bias y B son correctos para predecir al menos esta imagen de gato.

Ahora vemos la imagen del perro.

Cuando procesamos la imagen del perro nos dice que es 3.97 gato, 3.08 perro y 4.02 ave dándonos que el valor más grande es el valor predicho para el ave, mientras que la imagen de entrada se trata de la de un perro.

Por lo tanto no nos está diciendo, no nos está prediciendo correctamente que la imagen de entrada es un perro.

Y finalmente si procesamos la imagen del ave con los valores de W y bias seleccionados tenemos que el valor más grande también es el del ave, entonces está prediciendo de manera correcta que es un ave.

Sin embargo no son valores óptimos y principalmente lo vemos porque no está clasificando bien la imagen del perro.

Por lo tanto es muy importante tener una forma de cuantificar qué tan buenos son los pesos y bias actuales, qué tan buenos son los pesos W y bias B que tenemos actualmente.

Y para esto presentamos un nuevo concepto y este es un concepto de la función de una función de activación de alguna manera que es la función de activación que se utiliza más comúnmente para la última capa en problemas de clasificación multicláser.

La función de activación si recordamos del vídeo anterior, las redes neuronales necesitan para poder tener un poder de clasificación mejor, tener una función no lineal después de esta función lineal.

Hasta el momento estamos presentando un clasificador lineal, sin embargo necesitamos esta función de activación que nos permita determinar de mejor manera, de manera más intuitiva qué es lo que estamos representando por estos valores ó scores, estos puntajes.

Dado que es la última capa esta función de activación no está pensada en introducir esta no linealidad para incrementar la el poder de aproximación de la red normal.

Esta función de activación está hecha para dar más sentido a los resultados que estamos obteniendo y una de las formas que puede resultar más lógica para representar estos valores es que los convirtamos a una probabilidad y de tal forma que nos arroje en vez de darnos un valor numérico, un score que tal vez sea un tanto difícil de interpretar nos dé una probabilidad diciéndonos que en este caso la red neuronal está diciendo que nuestra salida es 40 por ciento posibilidades de ser un gato, 17 por ciento de ser un perro y 43 por ciento de ser un nave, pero dado que se trata de una imagen de un perro pues vemos que estos valores no son correctos necesitamos ajustar nuestros parámetros w y vallas.

Para obtener estos porcentajes vamos a utilizar la función softmax que es la que repito se utiliza más comúnmente como función de activación en la última capa de clasificadores multiclase.

Si entonces para clasificadores aún para clasificadores de redes neuronales multicapa con funciones con activaciones no lineales y incluso modelos más complejos como pueden ser redes neuronales convolucionales van a utilizar esta función softmax como activación.

Recapitulando este es el resultado son los scores que son el resultado de nuestra matriz w por x más nuestro vector y lo que vamos a hacer es esta implementar esta función softmax que es esta que estamos que estamos viendo aquí lo que nos indica es que interpretemos estos scores como probabilidades logarítmicas no normalizadas de cada una de las clases y realmente lo que esto significa es que vamos a tomar el exponencial de nuestros scores y esto es esta parte el numerador de esta expresión hacer la operación e el exponencial a este a este exponente 3.97 en este caso sería e a la 3.08 y aquí sería a la 4.02 así obtenemos estos valores y lo que vamos a hacer es dividirlas entre la suma de todos estos valores cada uno de ellos dividirla entre la suma que es el denominador de esta expresión matemática para así normalizar nuestros resultados y que todas las probabilidades sumen 1 si nos damos cuenta punto 40 más punto 17 más punto 43 efectivamente suman 1 entonces de aquí lo que vamos a hacer es cada uno de estos valores lo dividimos entre la suma de todos estos valores que es 130.55 y así tenemos que cada salida representa la probabilidad de que en el caso de la primera del primer elemento de la salida la probabilidad de que la imagen de entrada que se trata de un perro sea un gato la probabilidad de que sea un perro y luego la probabilidad de que sea un ave y el problema es que ahora sí nos damos cuenta que la probabilidad de que sea un perro es decir la clase correcta es muy baja por lo tanto necesitamos encontrar una forma de mejorar esta salida ya sabemos ya indicamos cómo está nuestra probabilidad y ahora lo que nos hace falta es medir qué tan buenos son los parámetros que arrojaron esta probabilidad y podemos darnos cuenta intuitivamente que que no son muy buenos porque están arrojando la probabilidad más baja para la clase correcta para la clase perro entonces cómo medimos esto cómo implementamos una función matemática una fórmula que nos ayude a cuantificar qué tan buenos son nuestros parámetros actuales y una vez que cuantifiquemos esto que tengamos una forma de hacerlo esto lo vamos a hacer mediante una función de pérdida o los function y lo que queremos es minimizar esta función porque esta función nos indica qué tan grande es nuestro error en este vídeo vimos cómo implementar la función de activación softmax para que los valores obtenidos a partir de una transformación lineal sean interpretados como probabilidades de que determinada entrada pertenezca a una clase en particular con esto lo que nos falta lo que necesitamos hacer es cómo podemos evaluar la calidad de los parámetros de nuestra red neuronal y con esto me refiero a qué tan buenos son es decir si vemos que la probabilidad de que la imagen pertenezca a un gato y efectivamente es un gato es muy baja podemos de manera intuitiva saber que es que hay algo malo en nuestros parámetros y que no son óptimos que estos puede ser ajustados pero para esto necesitamos poder evaluar de forma matemática qué tan óptimos son nuestros parámetros para esto en el siguiente vídeo vamos a presentar la función de pérdida que nos permite realizar este tipo de evaluación continuamos [SILENCIO] [MÚSICA]