Redes Neuronales Monocapa II

Modelos de la computación (Aprendizaje supervisado)

Francisco Fernández Navarro

Departamento de Lenguajes y Ciencias de la Computación
Área: Ciencias de la Computación e Inteligencia Artificial

Índice de contenidos

Regla de aprendizaje normalizada

Teorema

Partiendo de un vector de pesos normalizado, todos los pesos que se obtienen estarán normalizados.

Demostración

$$
\begin{aligned}
||\mathbf{w}(i,n+1)||^2 &= \sum_{k=1}^{K+1} w_k(i,n+1)^2 \\
&= \sum_{k=1}^{K+1} w_k(i,n)^2 + \left(2\frac{\sum_{k=1}^{K+1} w_k(i,n)(x_{nk})}{\sum_{k=1}^{K+1}(x_{nk})^2}\right)^2 \sum_{k=1}^{K+1}(x_{nk})^2 \\
&\quad -4\left(\frac{\sum_{k=1}^{K+1} w_k(i,n)(x_{nk})}{\sum_{k=1}^{K+1}(x_{nk})^2}\right)\sum_{k=1}^{K+1} w_k(i,n)(x_{nk}) \\
&= \sum_{k=1}^{K+1} w_k(i,n)^2 = 1.
\end{aligned}
$$

Regla de aprendizaje normalizada

Cuestión:

¿Os acordais de la ecuación siguiente

$$\eta^\star = \frac{\left|\sum_{k=1}^{K+1} w_k(i,n)(x_{nk})\right|}{\sum_{k=1}^{K+1}(x_{nk})^2}$$

en el que se asumía que $\left|\sum_{k=1}^{K+1} w_k(i,n)(x_{nk})\right| = \left|\sum_{k=1}^{K+1} w_k^\star(x_{nk})\right|$?

¿A que ayuda esa suposición?

A que $||\mathbf{w}(i,n)||^2 = 1, \forall i = 2, \ldots, I$, siempre que el vector ha sido inicializado de forma que $||\mathbf{w}(1,n)||^2 = 1$.

Regla de aprendizaje normalizada

Recordatorio

Veamos cómo generar $\mathbf{w}(1,0)$, de tal forma que $||\mathbf{w}(1,0)||^2 = 1$. En primer lugar inicializamos aleatoriamente el vector $\mathbf{w}(1,0)$:

$$\mathbf{w}(1,0) = \begin{bmatrix} w_2(1,0) = 2 \times rand() - 1 \\ w_2(1,0) = 2 \times rand() - 1 \\ \vdots \\ w_{K+1}(1,0) = 2 \times rand() - 1 \end{bmatrix}$$

Luego, normalizas el vector dividiéndolo por su norma (magnitud):

$$\mathbf{w}(1,0) = \frac{\mathbf{w}(1,0)}{\|\mathbf{w}(1,0)\|},$$

donde $\|\mathbf{w}(1,0)\|$ representa la norma euclidiana (magnitud) del vector $\mathbf{w}(1,0)$.

Índice de contenidos

Interpretando regla de aprendizaje del perceptrón simple

- En primer lugar definimos el vector $\mathbf{a}(i,n)$ (iteración i, patrón n, y dimensión K), como:

$$\mathbf{a}(i,n) = \begin{cases} \mathbf{x_n} & \text{si} \quad y_n = 1 \\ -\mathbf{x_n} & \text{si} \quad y_n = -1 \end{cases}$$

- Si asumimos una tasa de aprendizaje de 0.5, $\eta = 0.5$, nos queda que

$$\mathbf{w}(i,n+1) = \begin{cases} \mathbf{w}(i,n) + \mathbf{a}(i,n) & \text{si } (\mathbf{a}(i,n))'\mathbf{w}(i,n) \leq 0 \\ \mathbf{w}(i,n) & \text{en otro caso} \end{cases}$$

  con $\mathbf{w}(i,n) = (w_1(i,n), \ldots, w_{K+1}(i,n))$.
- Se realizan las correcciones siempre y cuando se producen clasificaciones incorrectas:

$$(\mathbf{a}(i,n))'\mathbf{w}(i,n) \leq 0$$

Interpretando regla de aprendizaje del perceptrón simple

- La regla de aprendizaje del Perceptrón intenta encontrar una solución para el siguiente sistema de desigualdades

$$(\mathbf{a}(i,n))'\mathbf{w}(i,n) > 0, \forall n = 1, \dots, N.$$

ya que en ese caso todos los patrones estarían clasificados correctamente.

- Función de optimización:

$$E(\mathbf{w}) = -\sum_{l \in L} (\mathbf{a}(i,l))'\mathbf{w}(i,l)$$

donde L es el conjunto de patrones clasificados incorrectamente utilizando el vector de pesos sinápticos $\mathbf{w}$.

Índice de contenidos

Estimación por lotes

- En el modelo por lotes actualizamos el vector de pesos sinápticos después de evaluar las salidas de la red para todos los patrones de entrenamiento. En este problema evitaremos usar el iterador sobre n, ya que no modificamos patrón a patrón.
- Metodo del descenso del gradiente:

$$\nabla E(\mathbf{w}) = -\sum_{l \in L} (\mathbf{a}(i,l))$$

- Estimación del vector de parámetros:

$$\begin{aligned} \mathbf{w}(i+1) &= \mathbf{w}(i) - \eta \nabla E(\mathbf{w}) \\ &= \mathbf{w}(i) + \eta \sum_{l \in L} (\mathbf{a}(i,l)) \end{aligned}$$

Flujo algorítmico - Algoritmo por lotes

PERCEPTRON-POR LOTES ($\mathcal{D}$, $\eta > 0$):

$\mathbf{X} \leftarrow (\mathbf{X} \;\; -\mathbf{1}_N) \in \mathbb{R}^{N\times(K+1)}$
for $k = 1$ until $K + 1$ **do**
    $w_k(0) \leftarrow 2 \times \mathrm{rand}() - 1, 0 \leq \mathrm{rand}() < 1$, $(\mathbf{w}(1) = (w_1(0), \ldots, w_{K+1}(0)))$.
end for
for $i = 1$ until I **do**
    $\Delta\mathbf{w}(i) \leftarrow \mathbf{0}_{K+1}$
    for $n = 1$ until N **do**
        $\hat{y}_n(i) \leftarrow \mathrm{sgn}(\sum_{k=1}^{K+1} w_k(i,n)x_{nk})$
        if error clasifi. **then**
            $\Delta\mathbf{w}(i) \leftarrow \Delta\mathbf{w}(i) + \mathbf{a}(i,n)$
        end if
    end for
    $\Delta\mathbf{w}(i) \leftarrow \eta\Delta\mathbf{w}(i)$
    $\mathbf{w}(i) \leftarrow \mathbf{w}(i-1) + \Delta\mathbf{w}(i)$.
end for

Índice de contenidos

La regla del bolsillo

Consiste en tener en cuenta el número de iteraciones consecutivas del algoritmo de perceptrón en las cuales no se ha modificado el vector de pesos sinápticos (para cada uno de los vectores que va generando), es decir, tener en cuenta el número de patrones que se han clasificado correctamente con dicho vector hasta que se ha encontrado el primer patrón que clasifica incorrectamente. Se tiene "guardado en el bolsillo" la mejor solución explorada, es decir, el vector de pesos sinápticos generado que ha conseguido, hasta el momento, el mayor número de iteraciones sin ser modificado. Cuando se encuentra un nuevo vector de pesos sinápticos que consigue un mayor número de clasificaciones correctas consecutivas que el que hay en el bolsillo entonces el vector del bolsillo se reemplaza por este. La solución final viene dada por el vector de pesos sinápticos guardado en el bolsillo.

Índice de contenidos

Ejercicio 1: Enunciado

Enunciado

Diseña una neurona que calcule la función AND con dos entradas.

Ejercicio 1: Solución

Solución

Podríamos diseñar nuestro perceptrón de la siguiente manera:

- Entradas: Necesitamos un perceptrón simple con 2 entradas ($K = 2$).
- Pesos Sinápticos: $w_1 = w_2 = 1$.
- Umbral: θ, que podría ser igual o mayor a 2.
- Función de Activación: Función signo, que emite "1" si la suma ponderada de las entradas es mayor o igual al umbral y "-1" en caso contrario.

La estimación del modelo para un patrón $\mathbf{x}_n$ sería:

$$\hat{y}(\mathbf{x}_n) = \mathrm{sgn}\left(\sum_{k=1}^{2} x_{nk} - 2\right)$$

Ejercicio 2: Enunciado

Enunciado

Diseña una neurona que calcule la función OR con dos entradas.

Ejercicio 2: Solución

Solución

Podríamos diseñar nuestro perceptrón de la siguiente manera:

- Entradas: Necesitamos un perceptrón simple con 2 entradas ($K = 2$).
- Pesos Sinápticos: $w_1 = w_2 = 1$.
- Umbral: θ, que podría ser igual o mayor a 1.
- Función de Activación: Función signo, que emite "1" si la suma ponderada de las entradas es mayor o igual al umbral y "-1" en caso contrario.

La estimación del modelo para un patrón $\mathbf{x}_n$ sería:

$$\hat{y}(\mathbf{x}_n) = \operatorname{sgn}\left(\sum_{k=1}^{2} x_{nk} - 1\right)$$

Ejercicio 3: Enunciado

Enunciado

Diseñar un perceptrón simple para clasificar a los alumnos que pueden acceder a una beca de investigación. Los alumnos aptos de una beca de investigación deberán tener una nota media superior a 1.5 de las 40 asignaturas cursadas por los alumnos. Las calificaciones de las asignaturas se puntúan con 0 (suspenso), 1 (aprobado), 2 (notable), 3 (sobresaliente) y 4 (matrícula de honor). Determinar el número de neuronas, así como los pesos sinápticos y umbrales de la red neuronal propuesta.

Ejercicio 3: Solución

Solución

Podríamos diseñar nuestro perceptrón de la siguiente manera:

- Entradas: Necesitamos un perceptrón simple con 40 entradas ($K = 40$).
- Pesos Sinápticos: $w_1, w_2, ..., w_K$, donde cada peso w_k es igual a $1/40$.
- Umbral: θ, que podría ser igual o mayor a 1.5.
- Función de Activación: Función signo, que emite “1” si la suma ponderada de las entradas es mayor o igual al umbral y “-1” en caso contrario.

La estimación del modelo para un patrón $\mathbf{x}_n$ sería:

$$\hat{y}(\mathbf{x}_n) = \operatorname{sgn}\left(\sum_{k=1}^{40} \frac{1}{40} x_{nk} - 1.5\right)$$

Ejercicio 4: Enunciado

Enunciado

Dados los pesos sinápticos y el umbral de un perceptrón simple con la arquitectura que se muestra mas abajo, debe determinarse la función booleana que implementa, teniendo en cuenta que la función de activación es la función signo. Los pesos sinápticos y el umbral son:

Entrada 1	2.58
Entrada 1	-2.31
Sesgo	3.18

Ejercicio 4: Solución

Solución

E1	E2	Potencial sináptico	$\hat{y}(\mathbf{x}_n)$
-1	-1	2.58 (-1) + (-2.31) (-1) - 3.18 =–0.27 – 3.18 = –3.45	-1
-1	1	2.58 (-1) + (-2.31) (1) - 3.18 =–4.89– 3.18 = –8.07	-1
1	-1	2.58 (1) + (-2.31) (-1) - 3.18 =4.89– 3.18 = 1.71	1
1	1	2.58 (1) + (-2.31) (1) - 3.18 =0.27– 3.18 = –2.91	-1

Ejercicio 5: Enunciado

Enunciado

Determinar los pesos sinápticos, umbral y función de activación de un perceptrón simple que indique si un número binario tiene la mayoría de sus bits activados o no.

Ejercicio 5: Solución

Solución

Podríamos diseñar nuestro perceptrón de la siguiente manera:

- Entradas: K bits (por ejemplo, para un número binario de 8 bits, tendríamos 8 entradas).
- Pesos Sinápticos: $w_1, w2, ..., w_K$, donde cada peso w_k es igual a 1.
- Umbral: θ, que podría ser igual o mayor a $K/2$.
- Función de Activación: Función signo, que emite "1" si la suma ponderada de las entradas es mayor o igual al umbral y "-1" en caso contrario.

La estimación del modelo para un patrón $\mathbf{x}_n$ sería:

$$\hat{y}(\mathbf{x}_n) = \operatorname{sgn}\left(\sum_{k=1}^{K} x_{nk} - \frac{K}{2}\right)$$

Enunciado

Diseña un perceptrón simple para determinar si un correo electrónico es spam o no. Los correos electrónicos se representarán mediante vectores de características binarias, donde cada característica representa la presencia (1) o ausencia (0) de ciertas palabras clave que son comunes en el spam. Se proporcionan tres características principales: "oferta", "ganador" y "urgente". Para que un correo electrónico se considere spam, debe contener al menos dos de estas palabras clave.

Ejercicio 6: Solución

Solución

Podríamos diseñar nuestro perceptrón de la siguiente manera:

- Entradas: Tendremos tres entradas binarias para representar las características “oferta”, ”ganador” y “urgente” del correo electrónico.
- Pesos Sinápticos: $w_1, w2, w_3$, donde cada peso w_k es igual a 1.
- Umbral: θ, que podría ser igual o mayor a 1.5.
- Función de Activación: Función signo, que emite “1” si la suma ponderada de las entradas es mayor o igual al umbral y “-1” en caso contrario.

La estimación del modelo para un patrón $\mathbf{x}_n$ sería:

$$\hat{y}(\mathbf{x}_n) = \operatorname{sgn}\left(\sum_{k=1}^{K} x_{nk} - 1.5\right)$$

Ejercicio 7: Enunciado

Enunciado

Diseña un perceptrón para clasificar a los candidatos que pueden ser admitidos en un programa de maestría en una prestigiosa universidad. Los candidatos elegibles deben cumplir con los siguientes criterios:

- Deben tener una licenciatura con una calificación promedio superior a 3.5 en una escala de 0 a 4.0.
- Deben tener al menos 3 años de experiencia laboral relacionada con el campo de estudio.
- Deben haber obtenido una puntuación mínima de 160 en el examen estandarizado de aptitud para la maestría.

Ejercicio 7: Enunciado

Solución

Vamos a resolverlo con dos capas:

- Primera capa (tres neuronas):
  - Primera neurona: Entrada x_1, su peso es 1, y el umbral $\theta_1 = 3.5$.
  - Segunda neurona: Entrada x_2, su peso es 1, y el umbral $\theta_2 = 3$.
  - Tercera neurona: Entrada x_3, su peso es 1, y el umbral $\theta_3 = 160$.
- Segunda capa (una neurona):
  - Entradas: Las 3 salidas de las neuronas de la capa anterior.
  - Pesos sinápticos: $w_1 = w_2 = w_3 = 1$
  - Umbral: $\theta = 2.5$

La estimación del modelo para un patrón $\mathbf{x}_n$ sería:

$$\hat{y}(\mathbf{x}_n) = \operatorname{sgn}\left(\sum_{l=1}^{3} s_l(\mathbf{x}_n) - 2.5\right), s_l(\mathbf{x}_n) = \operatorname{sgn}(x_{nl} - \theta_l), l \in \{1, 2, 3\}.$$

¡Gracias por vuestra atención!