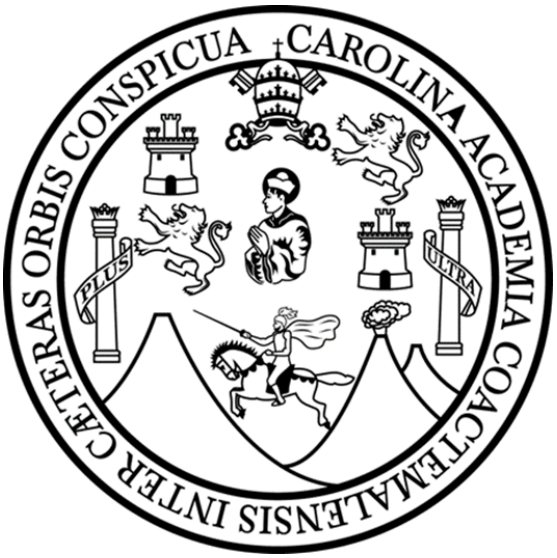
Universidad de San Carlos de Guatemala

Centro Universitario de Occidente

Ingeniería en Ciencias y Sistemas

Laboratorio Inteligencia Artificial 1



Explicación Matemática

202031288 Jhony Roel Fuentes Lopez

Explicación Matemática

1. Cargar y normalización de datos.

El dataset "Breast Cancer Wisconsin" tiene 569 muestras con 30 características cada una, más una etiqueta (0 para benigno, 1 para maligno) y una fila de Identificador. Una fila del archivo `wdbc.data` se ve así (primera muestra):

```
842302,M,17.99,10.38,122.80,1001.0,0.11840,...,0.11890
```

Normalización

Normalizamos los datos para que cada característica esté en el rango [0, 1]. La fórmula de normalización para una dato es:

$$X_{\text{norm},j} = \frac{X_j - X_{j,\min}}{X_{j,\max} - X_{j,\min}}$$

Donde:

$Xj\ min\ =\ el\ valor\ mínimo\ dentro\ de\ la\ columna\ seleccionada$

$Xj\ max\ =\ el\ valor\ máximo\ dentro\ de\ la\ columna\ seleccionada$

Ejemplo con "texture1" (índice 1):

El valor original es 10.38, dentro de estos datos el valor más alto es: 39.28 y el más pequeño es: 9.71, entonces 10.38 normalizado quedaría:

$$\frac{10.38 - 9.71}{39.28 - 9.71} = \frac{0.67}{29.57} \approx 0.02266$$

Inicialización del Perceptrón

El perceptrón se inicia con Un vector de tamaño 2 (porque usamos 2 características), lo iniciamos aleatoriamente entre [-1, 1]. Por ejemplo:

$$\mathbf{w} = [w_0, w_1] = [0.5, -0.3]$$

Sesgo b: Un valor aleatorio entre [-1, 1], por ejemplo:

$$b = 0.2$$

Tasa de aprendizaje n: Definida por el usuario, por ejemplo:

$$\eta = 0.01$$

3. Predicción del Perceptrón

El método `activación` calcula la predicción utilizando la función Función de Escalón de Heaviside La ecuación es:

$$z = \sum w_j x_j + b$$

Y la predicción es:

$\widehat{y}$ = { 1 si z +b >= 0 , 0 si z+b < 0}

En el código, esto se implementa como:

$$z = \mathbf{w} \cdot \mathbf{x} = w_0 x_0 + w_1 x_1$$

Ejemplo

Supongamos que X = [0.524,0.02266] valores normalizados de radius1 y texture1
Pesos: w=[0.5,−0.3], sesgo: b=0.2

entonces tendríamos.

$$z = (0.5 \cdot 0.524) + (-0.3 \cdot 0.02266) = 0.262 - 0.006798 = 0.255202$$

$$z + b = 0.255202 + 0.2 = 0.455202$$

Como z+b > 0, entonces tendríamos una predicción de 1 (maligno)

Cálculo del Error

El error para una muestra se calcula como:

$$\text{error} = y - \hat{y}$$

donde:

y = valor verdadero

$\widehat{y}$ = predicción

En el ejemplo anterior:

y = 1 $\widehat{y}$ = 1

error = 1 - 1 = 0

Si la predicción falla entonces tendríamos:

$$\text{error} = 1 - 0 = 1$$

El error total por época se acumula como:

$$\text{total}_\text{e}\text{rror} = \sum (\text{error}_i)^2$$

Actualización de Pesos y Sesgo

Los pesos y el sesgo se actualizan usando la regla de aprendizaje del perceptrón:

$$w_j \leftarrow w_j + \eta \cdot x_j \cdot \text{error}$$

$$b \leftarrow b + \eta \cdot \text{error}$$

Ejemplo

-Usando los valores anteriores

- Pesos iniciales: w = [0.5, -0.3], sesgo: b = 0.2

$w_0 \leftarrow 0.5 + 0.01 \cdot 0.524 \cdot 0 = 0.5$

$w_1 \leftarrow -0.3 + 0.01 \cdot 0.02266 \cdot 0 = -0.3$

$b \leftarrow 0.2 + 0.01 \cdot 0 = 0.2$

En este caso, como no existe error entonces los sesgos y pesos no cambian.

Si error = 1 entonces:

$w_0 \leftarrow 0.5 + 0.01 \cdot 0.524 \cdot 1 = 0.5 + 0.00524 = 0.50524$

$w_1 \leftarrow -0.3 + 0.01 \cdot 0.02266 \cdot 1 = -0.3 + 0.0002266 = -0.2997734$

$b \leftarrow 0.2 + 0.01 \cdot 1 = 0.21$

Este proceso se repite para cada muestra y cada época.

7. Cálculo de la Exactitud

Para calcular la exactitud en los datos de prueba, los datos de prueba son los que no se incluyen dentro de los datos de entrenamiento, hacemos predicciones para cada muestra y las comparamos con las etiquetas reales

predicciones = [activacion(X_i) para cada X_i en datosPrueba]

Dividimos los aciertos entre el número de datos que se usaron para la prueba.

Ejemplo:

Supongamos que dataTest = 114 (20% del dataset).

Si 90 predicciones son correctas:

$$\text{exactitud} = \frac{90}{114} \approx 0.7895 \ (78.95\%)$$

Análisis de los resultados (efecto de η, convergencia, limitaciones).

Efecto de n.

Efecto: Con un η grande, los ajustes a los pesos y al sesgo son más grandes en cada iteración, lo que acelera el aprendizaje.

Ventajas:

- Útil cuando el número de épocas es limitado y necesitas que el modelo aprenda rápidamente.

Valores Pequeños n

- **Efecto**: Cuando n es pequeño, los ajustes a los pesos y al sesgo son pequeños en cada iteración. Esto hace que el perceptrón aprenda de manera más lenta y suave.
- **Ventajas**:

- Mayor estabilidad en el aprendizaje, con menos riesgo de "saltos" grandes que puedan hacer que el modelo diverge.
- Puede converger a una solución más precisa si el número de épocas es suficiente.

Convergencia:

Los datos de error convergen casi siempre en un rango de 70-80, cuando %de datos de entrenamiento está entre 0.8 y 0.9 a partir de un número de épocas de 250 aproximadamente.

Para %de entrenamiento 0.1 - 0.2: converge en un rango de 5 -15 errores.

Para %de entrenamiento 0.4 - 0.6: converge en un rango de 50 y 60 errores con un número de épocas de 200.

Para %de entrenamiento 0.4 - 0.8: converge en un rango de 60 y 75 errores con un número de épocas de 200.

Limitaciones

Este tipo de perceptrón sólo puede encontrar una frontera de decisión lineal (una línea recta en 2D, un hiperplano en dimensiones superiores). Si los datos no son linealmente separables,al modelo le costará más encontrar valores óptimos , y la exactitud será baja.