Regresión Logística

Dr. Mauricio Toledo-Acosta

Diplomado Ciencia de Datos con Python

Table of Contents

Regresión Logística

Regresión Logística

El modelo logístico (o logit) es un modelo estadístico que modela la probabilidad de que se produzca un suceso haciendo que las probabilidades logarítmicas (log-odds) del suceso sean una combinación lineal de una o más variables independientes.

La regresión logística es un algoritmo para estimar los parámetros de un modelo logístico para resolver la tarea de clasificación.

Formulación matemática

Supongamos que tenemos datos $x_1, \ldots, x_n$ que pertenecen a dos clases clase_0 y clase_1. Queremos estimar

$$p(\text{clase}_j \mid x)$$

Formulación matemática

Supongamos que tenemos datos $x_1, \ldots, x_n$ que pertenecen a dos clases clase_0 y clase_1. Queremos estimar

$$p(\text{clase}_j \mid x)$$

Queremos que se satisfaga

$$p(\text{clase}_1 \mid x_i) \begin{cases} > 0.5, & y_i = 1 \\ < 0.5, & y_i = 0. \end{cases}$$

Formulación matemática

Supongamos que tenemos datos $x_1, \ldots, x_n$ que pertenecen a dos clases clase_0 y clase_1. Queremos estimar

$$p(\text{clase}_j \mid x)$$

Queremos que se satisfaga

$$p(\text{clase}_1 \mid x_i) \begin{cases} > 0.5, & y_i = 1 \\ < 0.5, & y_i = 0. \end{cases}$$

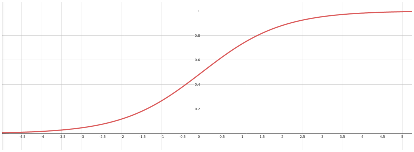
Denotamos $p(\text{clase}_1 \mid x_i)$ como $p(x_i)$. El problema se traduce en encontrar coeficientes β_0, β_1 tales que

$$\log\left(\frac{p(x)}{1 - p(x)}\right) = \beta_0 + \beta_1 x$$

¿Cómo obtenemos las predicciones?

Despejando, tenemos

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \in (0, 1).$$



Para parametros β_0, β_1 obtenemos probabilidades para cada dato x_i

$$p_i = p(x_i).$$

¿Cómo medimos la calidad de las predicciones?

Usamos la función de perdida logistic loss (binary cross-entropy o log-loss):

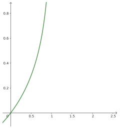
$$L = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(p_i) + (1 - y_i)\log(1 - p_i)\right) > 0$$

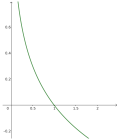
¿Cómo medimos la calidad de las predicciones?

Usamos la función de perdida logistic loss (binary cross-entropy o log-loss):

$$L = -\frac{1}{n}\sum_{i=1}^{n}(y_i \log(p_i) + (1-y_i)\log(1-p_i)) > 0$$

- Si x_i es de clase $y_i = 0$, la perdida en este punto es $-\log(1-p_i)$. Entre menor sea p_i, menor es el costo en el que se incurre
- Si x_i es de clase $y_i = 1$, la perdida en este punto es $-\log(p_i)$. Entre mayor sea p_i, menor es el costo en el que se incurre





Encontrar los parámetros usando la función de perdida

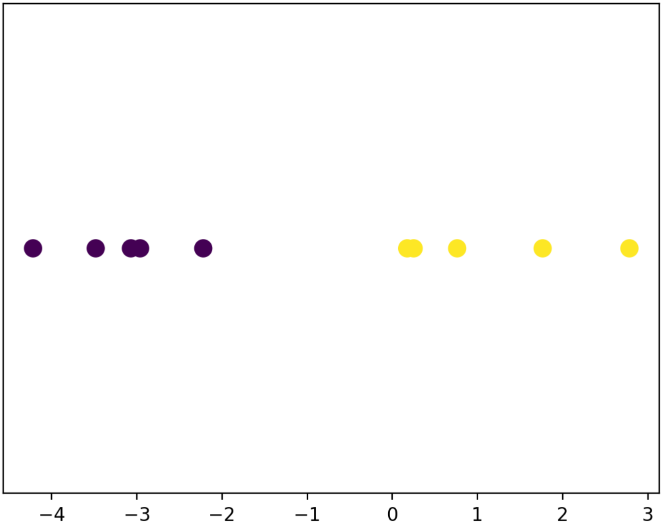
Usando la función de perdida, el problema se convierte en encontrar los parametros β_0, β_1 que minimizan la función log-loss.

Esto lo hacemos resolviendo

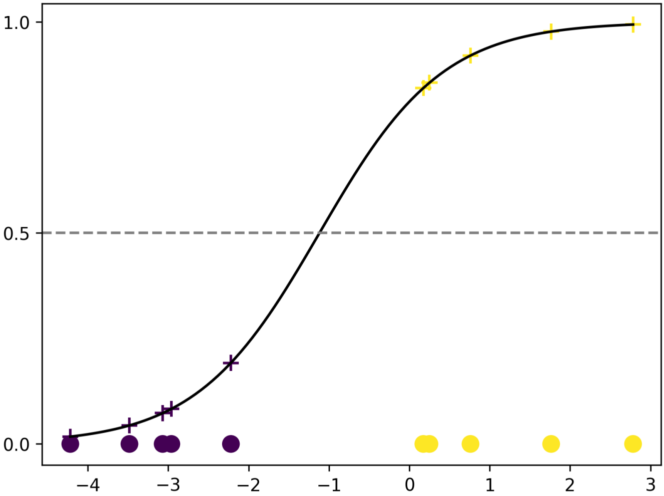
$$\frac{\partial}{\partial \beta_0} L = 0$$
$$\frac{\partial}{\partial \beta_1} L = 0$$

Para este fin se usan diversos métodos numéricos basados principalmente en descenso de gradiente.

Un Ejemplo



Un Ejemplo



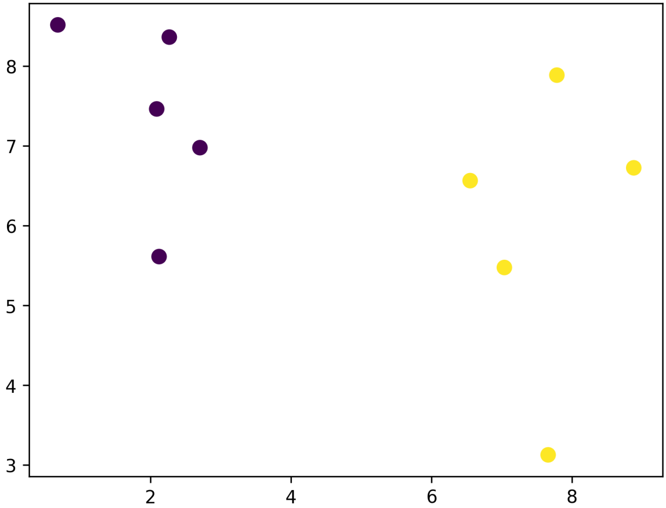
Clasificación multiclase

Si tenemos un problema de clasificación multiclase se usa un enfoque *one vs all*. Es decir, si tenemos m clases, estas están dadas por

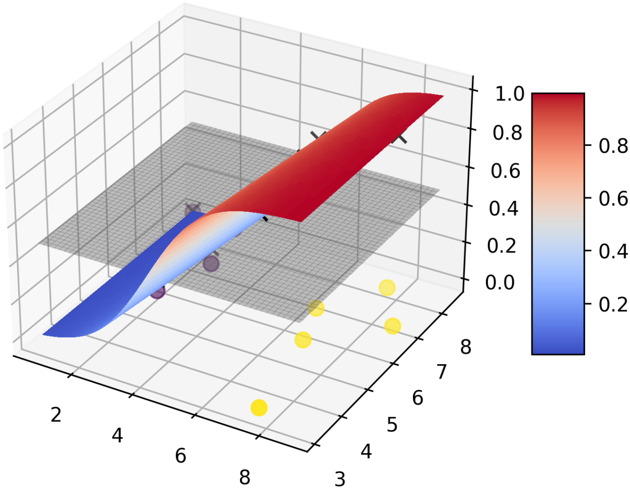
$$
\begin{array}{rcccc}
\text{clase 0:} & 1 & 0 & \cdots & 0 \\
\text{clase 1:} & 0 & 1 & \cdots & 0 \\
\dots & & & \dots & \\
\text{clase m:} & 0 & 0 & \cdots & 1
\end{array}
$$

Al realizar predicciones podemos obtener la predicción de la etiqueta, o las probabilidades de pertencia a cada clase.

Otro Ejemplo



Otro Ejemplo



Regularización

Si tenemos varias features, es decir, cada dato es de la forma $x = (x^{(1)}, ..., x^{(d)})$ el problema es

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \beta_1 x^{(1)} + \cdots + \beta_d x^{(d)}$$

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x^{(1)} + \cdots + \beta_d x^{(d)})}}$$

Regularización

Si tenemos varias features, es decir, cada dato es de la forma $x = (x^{(1)}, ..., x^{(d)})$ el problema es

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \beta_1 x^{(1)} + \cdots + \beta_d x^{(d)}$$

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x^{(1)} + \cdots + \beta_d x^{(d)})}}$$

Conforme crecen el número de features existe el riesgo del *overfitting*. El *overfitting* (sobreajuste) es un error de modelado en el cual se ajusta excesivamente un modelo a un conjunto de datos. Entonces, capta y aprendemos el ruido del conjunto de datos y podemos no ajustar a los nuevos datos entrantes.

Regularización

Si tenemos varias features, es decir, cada dato es de la forma $x = (x^{(1)}, ..., x^{(d)})$ el problema es

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \beta_1 x^{(1)} + \cdots + \beta_d x^{(d)}$$

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x^{(1)} + \cdots + \beta_d x^{(d)})}}$$

Conforme crecen el número de features existe el riesgo del *overfitting*. Para solucionar esto podemos descartar algunas features (*feature selection*) o hacer regularización.

Regularización

Consiste en modificar la función de costo añadiendo un término que penalicé la magnitud de los coeficientes.

- L1 (Lasso)

$$L = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right) + \frac{1}{C}\sum_{i=0}^{d}|\beta_i|$$

- L2 (Ridge)

$$L = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right) + \frac{1}{C}\sum_{i=0}^{d}\beta_i^2$$

La regularización es útil cuando tenemos un número grande de features. **Lasso puede eliminar la contribución de un subconjunto de ellas, mientras que Ridge puede atenuarlas.**

Parámetros Importantes

Aglunos de los hiperparámetros más importantes para la regresión logística son:

- `solver`: El algoritmo usado para resolver el problema de optimización.
- `fit_intercept`: Si usamos el intercepto β_0.
- `penalty`: Tipo de regularización.
- `C`: Inverso del coeficiente de regularización.

Ventajas y desventajas de la regresión logística

Ventajas:

- Coeficientes interpretables.
- Es un modelo simple, sin muchos parámetros para estimar.
- Es un modelo muy estudiado y entendido.
- Predicciones rápidas.

Desventajas:

- Sensible a outliers.
- No funciona con valores faltantes.
- Es sensible a features correlacionadas.
- No es un método del estado del arte.

Es buena idea usarlo como método *baseline*.