Classification problem

Logistic regression, decision boundary, overfitting and regularization

Khiem Nguyen

Email	`khiem.nguyen@glasgow.ac.uk`
MS Teams	`khiem.nguyen@glasgow.ac.uk`
Whatsapp	+44 7729 532071 (Emergency only)

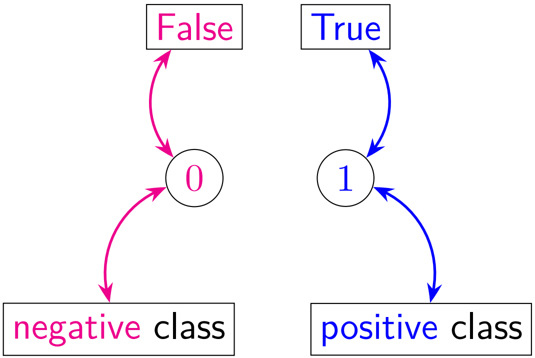
May 18, 2025

Table of Contents

Classification

Question	Answer
Is this email spam?	no yes
Is the transaction fraudulent?	no yes
Is the tumor malignant?	no yes

- y can only be one of two values
  false true
- "binary classification"
- class = category

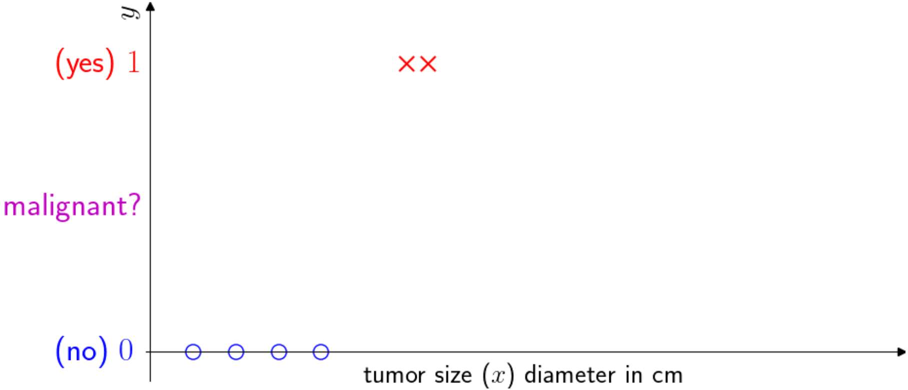


negative $\neq$ bad, positive $\neq$ good.

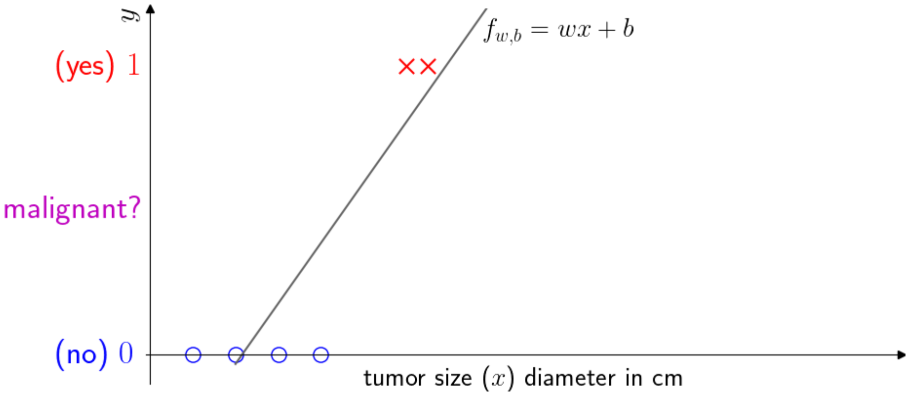
negative $\longleftrightarrow$ absence of a property

positive $\longleftrightarrow$ presence of a property

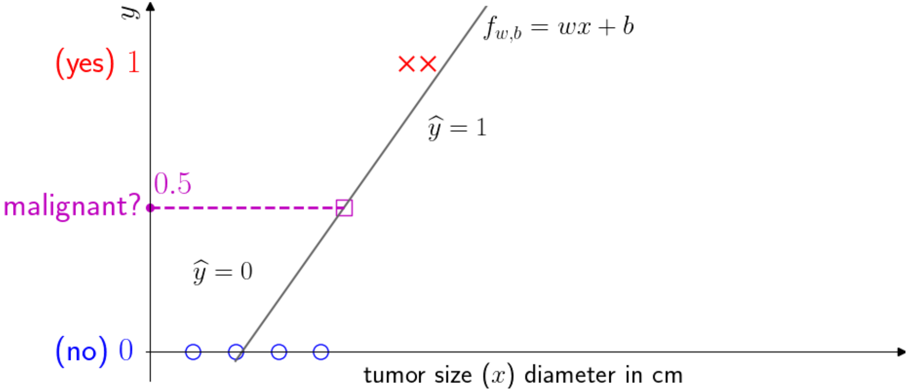
Decision boundary



Decision boundary



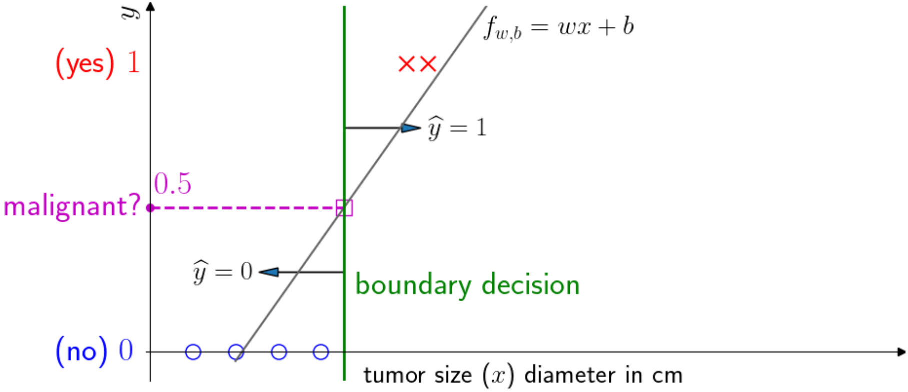
Decision boundary



$$\text{if } f_{w,b}(x) \leq 0.5 \quad \rightarrow \quad \hat{y} = 0$$

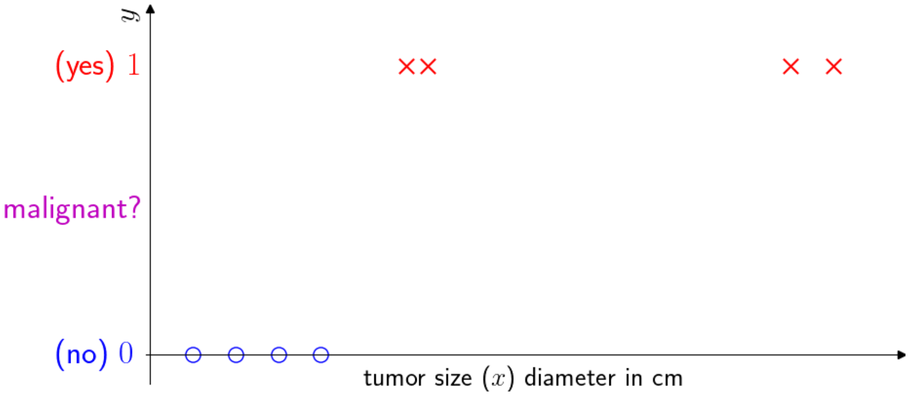
$$\text{if } f_{w,b}(x) < 0.5 \quad \rightarrow \quad \hat{y} = 1$$

Decision boundary

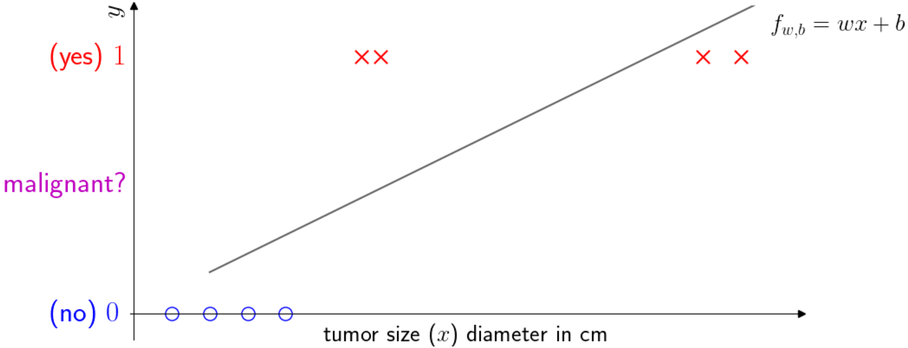


$$\text{if } f_{w,b}(x) \leq 0.5 \quad \rightarrow \quad \hat{y} = 0$$
$$\text{if } f_{w,b}(x) > 0.5 \quad \rightarrow \quad \hat{y} = 1$$

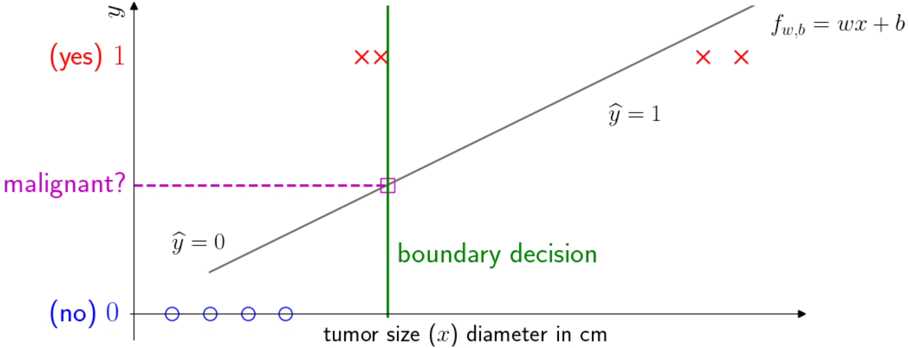
Decision boundary



Decision boundary



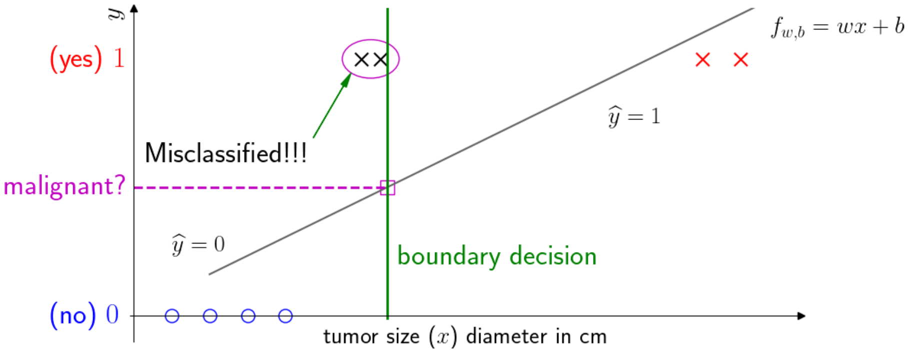
Decision boundary



if $f_{w,b}(x) \leq 0.5 \quad \rightarrow \quad \hat{y} = 0$

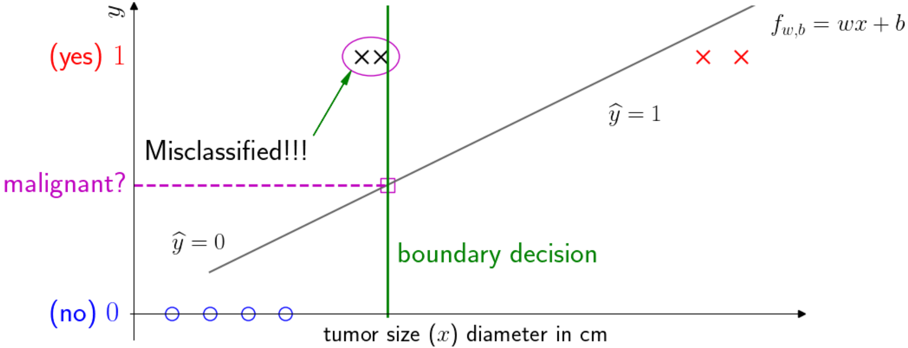
if $f_{w,b}(x) > 0.5 \quad \rightarrow \quad \hat{y} = 1$

Decision boundary



$$\text{if } f_{w,b}(x) \leq 0.5 \quad \rightarrow \quad \hat{y} = 0$$
$$\text{if } f_{w,b}(x) > 0.5 \quad \rightarrow \quad \hat{y} = 1$$

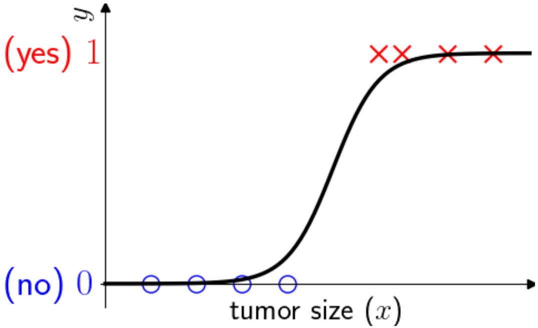
Decision boundary



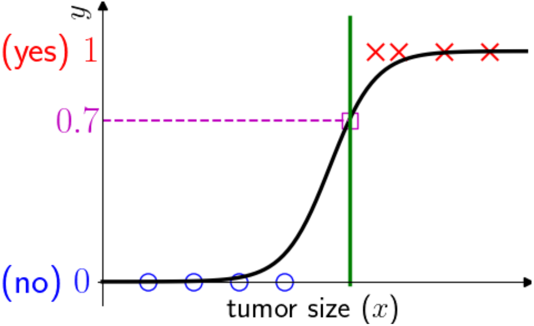
$$\text{if } f_{w,b}(x) \leq 0.5 \quad \rightarrow \quad \hat{y} = 0$$
$$\text{if } f_{w,b}(x) > 0.5 \quad \rightarrow \quad \hat{y} = 1$$

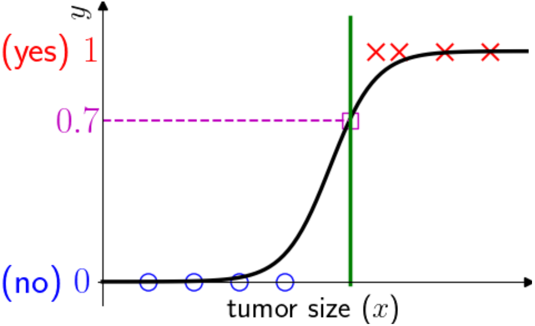
What to do: **logistic regression!!!**

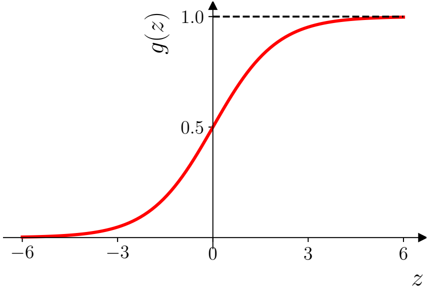
Logistic regression



Logistic regression





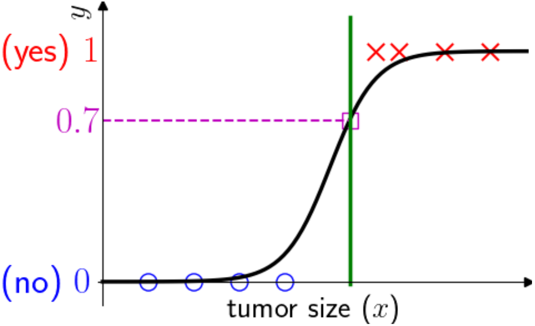


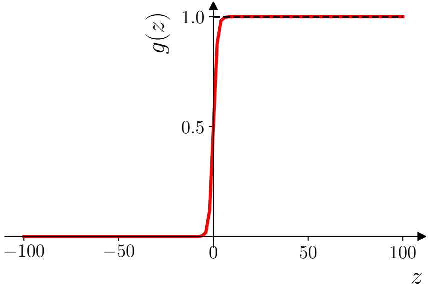
sigmoid function – logistic function

outputs between 0 and 1

$$g(z) = \frac{1}{1 + \mathrm{e}^{-z}} = \frac{1}{1 + \exp(-z)}, \quad 0 < g(z) < 1$$

Logistic regression





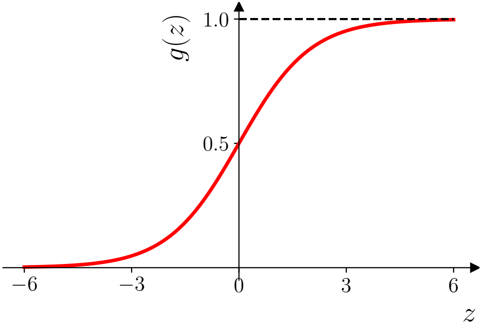
sigmoid function – logistic function

outputs between 0 and 1

$$g(100) \approx \frac{1}{1} = 1, \quad g(-100) \approx \frac{1}{\infty} = 0$$

$$g(\infty) = \frac{1}{1} = 1, \quad g(-\infty) = \frac{1}{+\infty} = 0.$$

Logistic regression



sigmoid function – logistic function
outputs between 0 and 1

$$g(z) = \frac{1}{1+\mathrm{e}^{-z}}, \quad 0 < g(z) < 1$$

$$\boxed{z = \vec{w} \cdot \vec{x} + b}$$

$$\downarrow$$

$$z$$

$$\downarrow$$

$$\boxed{g(z) = \frac{1}{1+\exp(-z)}}$$

$$f_{\vec{w},b}(\vec{x}) = g(\underbrace{\vec{w} \cdot \vec{x} + b}_{z}) = \frac{1}{1+\mathrm{e}^{-(\vec{w}\cdot\vec{x}+b)}}$$

$$= \frac{1}{1+\exp(-(\vec{w} \cdot \vec{x} + b))}$$

logistic regression!

Interpretation of logistic regression input

$$f_{w,b}(\vec{x}) = \frac{1}{1 + \mathrm{e}^{-(\vec{w}\cdot\vec{x}+b)}}$$

$$f_{\vec{w},b}(\vec{x}) = P(y = 1|\vec{x}; \vec{w}, b)$$

Interpretation: probability that class of $\vec{x}$ is 1

Probability that y is 1,

given input $\vec{x}$, parameters $\vec{w}, b$

Example:

- x is "tumor size"
- y is 0 (not malignant) or 1 (malignant)

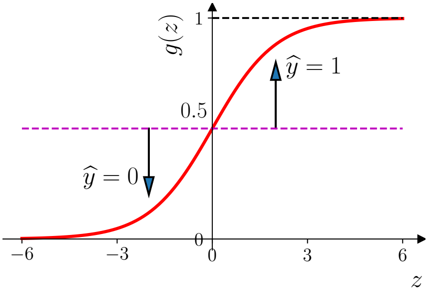
$$P(y = 0) + P(y = 1) = 1$$

$f_{\vec{w},b}(\vec{x}) = 0.7$ implies 70%

chance that y given $\vec{x}$ is 1.

$$\Rightarrow P(y = 0) = 1 - P(y = 1)$$

Decision boundary



$$f_{\vec{w},b}(\vec{x}) = g(\underbrace{\vec{w} \cdot \vec{x} + b}_{z}) = \frac{1}{1 + \mathrm{e}^{-(\vec{w} \cdot \vec{x} + b)}}$$
$$= P(y = 1 | x; \vec{w}, b)$$

0 or 1?: Is $f_{\vec{w},b}(\vec{x}) \geq 0.5$, $\quad 0.5 \rightarrow$ threshold

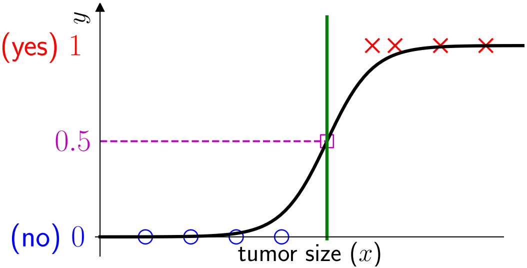
Yes: $\hat{y} = 1$ $\qquad$ No: $\hat{y} = 0$

When is $f_{\vec{w},b}(\vec{x}) \geq 0.5$?	$f_{\vec{w},b}(\vec{x}) \leq 0.5$?
$g(z) \geq 0.5$	$\cdots$
$z \geq 0$	$\cdots$
$\vec{w} \cdot \vec{x} + b \geq 0$	$\vec{w} \cdot \vec{x} + b \leq 0$
$\hat{y} = 1$	$\hat{y} = 0$

Decision boundary: 1D problem

- For single variable input, the decision boundary is

$$wx + b = 0 \quad \Leftrightarrow \quad x = -b/w$$



Question: What happen if we change the threshold from 0.5 to 0.7?

Decision boundary: 1D problem

- Denote the threshold τ with $0 < \tau < 1$:

$$
\begin{aligned}
& \frac{1}{1+\exp(-(\vec{w}\cdot\vec{x}+b))} = \tau \\
\Leftrightarrow \quad & \tau + \tau\exp(-(\vec{w}\cdot\vec{x}+b)) = 1 \\
\Leftrightarrow \quad & \tau\exp(-(\vec{w}\cdot\vec{x}+b)) = 1-\tau \\
\Leftrightarrow \quad & \exp(-(\vec{w}\cdot\vec{x}+b)) = \frac{1}{\tau} - 1 \\
\Leftrightarrow \quad & -\vec{w}\cdot\vec{x}+b = \log(\kappa), \quad \kappa = \frac{1}{\tau} - 1
\end{aligned}
$$

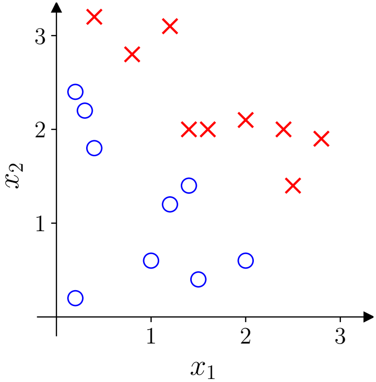
- For 1D problem:

$$
wx + b = \log(\kappa) \quad \Leftrightarrow \quad x = \frac{\log(\kappa) - b}{w}
$$

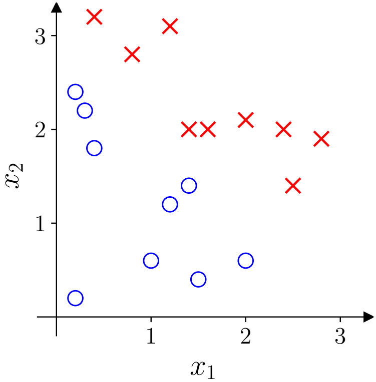
$\rightarrow$ a vertical straight line different from

$x = -b/w$ corresponding to the threshold $\tau = 0.5 \leftrightarrow \kappa = 1, \log(\kappa) = 0$

Decision boundary: 2D problem



Decision boundary: 2D problem



$$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1x_1 + w_2x_2 + b)$$

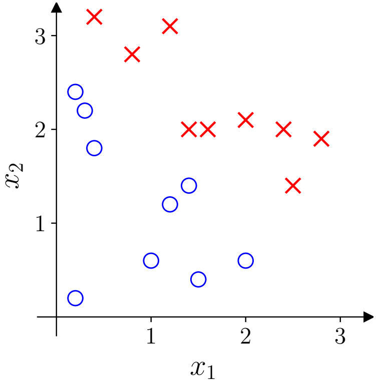
Decision boundary:

Assume the threshold $y_{\text{threshold}} = 0.5$

$$z = \vec{w} \cdot \vec{x} + b = 0 \quad \Leftrightarrow \quad w_1x_1 + w_2x_2 + b = 0$$

Clearly, this is just a line in 2D plane

Decision boundary: 2D problem



$$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1x_1 + w_2x_2 + b)$$

Decision boundary:

Assume the threshold $y_{\text{threshold}} = 0.5$

$z = \vec{w} \cdot \vec{x} + b = 0 \quad \Leftrightarrow \quad w_1x_1 + w_2x_2 + b = 0$

Clearly, this is just a line in 2D plane

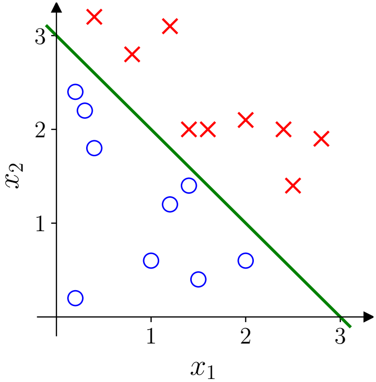
Assume after 'training': $w_1 = w_2 = 1, b = -3$
$\rightarrow$

$$w_1x_1 + w_2x_2 + b = 0$$

$$x_1 + x_2 - 3 = 0$$

$$x_1 + x_2 = 3$$

Decision boundary: 2D problem



$$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1x_1 + w_2x_2 + b)$$

Decision boundary:

Assume the threshold $y_{\text{threshold}} = 0.5$

$z = \vec{w} \cdot \vec{x} + b = 0 \quad \Leftrightarrow \quad w_1x_1 + w_2x_2 + b = 0$

Clearly, this is just a line in 2D plane

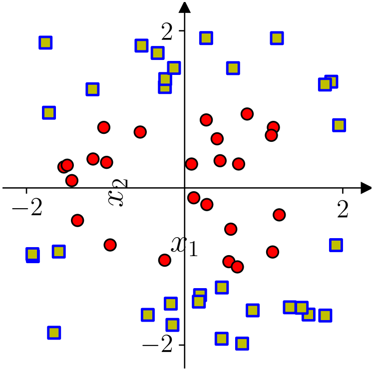
Assume after 'training': $w_1 = w_2 = 1, b = -3$
$\rightarrow$

$$w_1x_1 + w_2x_2 + b = 0$$

$$x_1 + x_2 - 3 = 0$$

$$x_1 + x_2 = 3$$

Nonlinear decision boundary



$$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1 x_1^2 + w_2 x_2^2 + b)$$

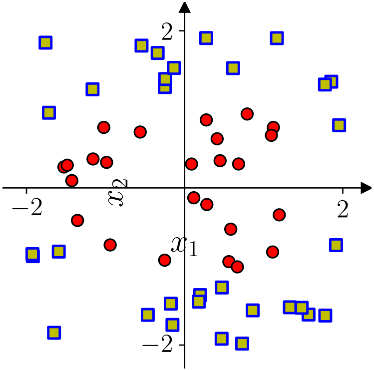
Decision boundary:

Assume the threshold $= 0.5$

$$z = w_1 x_1^2 + w_2 x^2 - 1 = 0$$

Ellipse in 2D plane

Nonlinear decision boundary



$$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1 x_1^2 + w_2 x_2^2 + b)$$

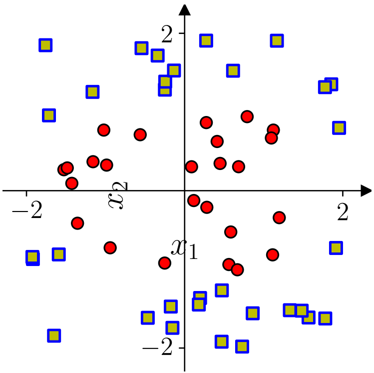
Decision boundary:

Assume the threshold $= 0.5$

$$z = w_1 x_1^2 + w_2 x^2 - 1 = 0$$

Ellipse in 2D plane

Nonlinear decision boundary



$$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1 x_1^2 + w_2 x_2^2 + b)$$

Decision boundary:

Assume the threshold $= 0.5$

$$z = w_1 x_1^2 + w_2 x^2 - 1 = 0$$

Ellipse in 2D plane

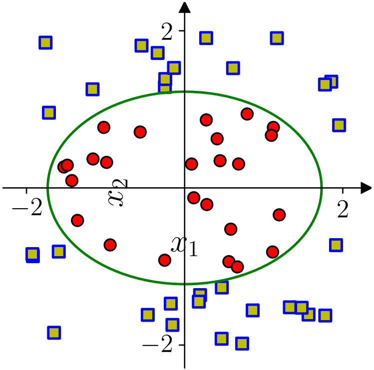
Assume after 'training':
$w_1 = 1, w_2 = 2, b = -3$

$$w_1 x_1 + w_2 x_2 + b = 0$$

$$x_1^2 + 2x_2^2 - 3 = 0$$

$$x_1^2 + 2x_2^2 = 3$$

Nonlinear decision boundary



$$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1 x_1^2 + w_2 x_2^2 + b)$$

Decision boundary:

Assume the threshold $= 0.5$

$$z = w_1 x_1^2 + w_2 x^2 - 1 = 0$$

Ellipse in 2D plane

Assume after 'training':
$w_1 = 1, w_2 = 2, b = -3$

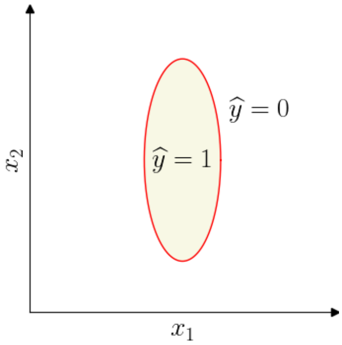
$$w_1 x_1 + w_2 x_2 + b = 0$$

$$x_1^2 + 2x_2^2 - 3 = 0$$

$$x_1^2 + 2x_2^2 = 3$$

Nonlinear decision boundary

The decision boundary can be as wiggly and complex as we may want.



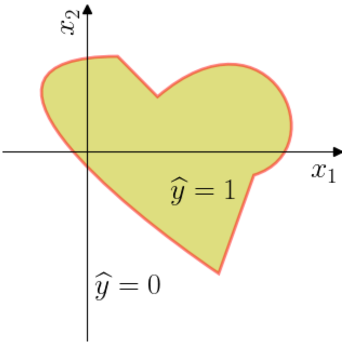


Table of Contents

Training set

tumor size (cm) x_1	$\cdots$	age x_n	malignant? y
10		52	1
2		73	0
5		55	0
12		49	1
$\vdots$		$\vdots$	$\vdots$

Training set

tumor size (cm) x_1	$\cdots$	age x_n	malignant? y
10		52	1
2		73	0
5		55	0
12		49	1
$\vdots$		$\vdots$	$\vdots$

$i = 1, \dots, m \quad \longleftarrow$ training examples

$j = 1, \dots, n \quad \longleftarrow$ features

target y is 0 or 1

Model function

$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1 + \mathrm{e}^{-(\vec{w}\cdot\vec{x}+b)}}$$

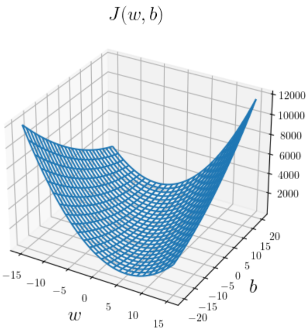
Squared error cost

$$J(\vec{w}, b) = \frac{1}{m}\sum_{i=1}\frac{1}{2}(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})^2 = \frac{1}{m}\sum_{i=1}^{m}\underbrace{L\big[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\big]}_{\text{loss (the example } (i))}$$

Squared error cost

$$J(\vec{w}, b) = \frac{1}{m} \sum_{i=1} \frac{1}{2}(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})^2 = \frac{1}{m} \sum_{i=1}^{m} \underbrace{L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right]}_{\text{loss (the example } (i))}$$

$$f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b$$

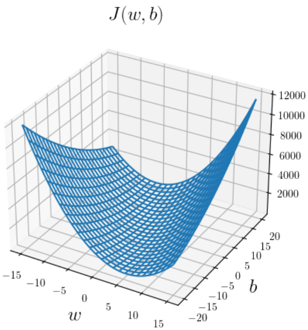


Convex cost function

Squared error cost

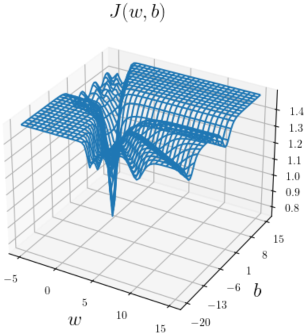
$$J(\vec{w},b) = \frac{1}{m}\sum_{i=1} \frac{1}{2}(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})^2 = \frac{1}{m}\sum_{i=1}^{m} \underbrace{L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right]}_{\text{loss (the example } (i))}$$

$$f_{\vec{w},b}(\vec{x}) = \vec{w}\cdot\vec{x} + b$$



Convex cost function

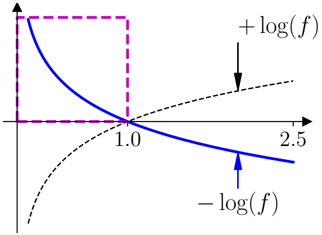
$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1+\mathrm{e}^{-(\vec{w}\cdot\vec{x}+b)}}$$



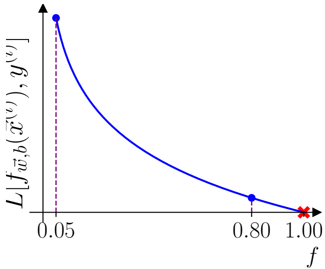
Non-convex cost function

Logistic loss function

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = \begin{cases} -\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$



Loss is lowest when $f_{\vec{w},b}(x^{(i)}))$ predicts close to true label $y^{(i)}$ and highest when f predicts far from true label.

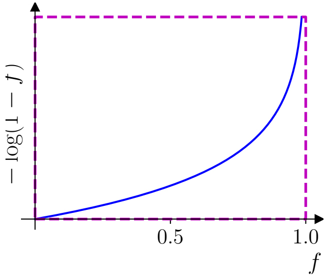


As $f_{\vec{w},b}(x^{(i)}) \to 0$, then loss $\to \infty$ ☹☹☹

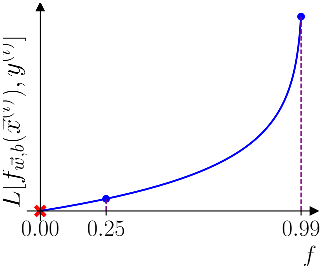
As $f_{\vec{w},b}(x^{(i)}) \to 1$, then loss $\to 0$ ☺☺☺

Logistic loss function

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = \begin{cases} -\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$



Loss is lowest when $f_{\vec{w},b}(x^{(i)}))$ predicts close to true label $y^{(i)}$ and highest when f predicts far from true label.



As $f_{\vec{w},b}(x^{(i)}) \to 1$, then loss $\to \infty$ ☹☹☹

As $f_{\vec{w},b}(x^{(i)}) \to 0$, then loss $\to 0$ ☺☺☺

Cost function for logistic regression

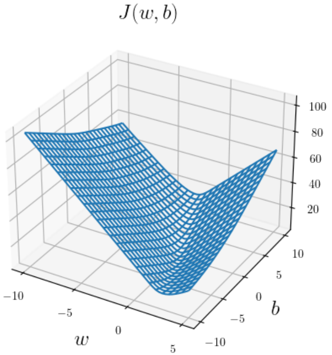
$$J(\overrightarrow{w}, b) = \frac{1}{m}\sum_{i=1}^{m} L[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{i}]$$

$$L[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{(i)}] = \begin{cases} -\log\left(f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$

Cost function for logistic regression

$$J(\overrightarrow{w}, b) = \frac{1}{m}\sum_{i=1}^{m} L[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{i}]$$

$$L[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{(i)}] = \begin{cases} -\log\left(f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$



Cost function for logistic regression

$$J(\overrightarrow{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{i}]$$

$$L[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{(i)}] = \begin{cases} -\log\left(f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$

Cost function $J(\overrightarrow{w}, b)$ is convex $\rightarrow$ can reach a global minimum

Find $\overrightarrow{w}, b$: Minimize $J(\overrightarrow{w}, b)$ with respect to $\overrightarrow{w}, b$

$$\boxed{\min_{\overrightarrow{w},b} J(\overrightarrow{w}, b)}$$

Simplified cost function

$$L\left[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = \begin{cases} -\log\left(f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$

Simplified cost function

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = \begin{cases} -\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$

$$\Rightarrow \quad L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = -y^{(i)} \log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) - (1 - y^{(i)}) \log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)$$

Simplified cost function

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = \begin{cases} -\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$

$$\Rightarrow \quad L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = -y^{(i)} \log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) - (1 - y^{(i)}) \log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)$$

- If $y^{(i)} = 1$

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = -1 \times \log(f_{\vec{w},b}(\vec{x}^{(i)})) - (1-1) \times \smiley = -\log(f)$$

Simplified cost function

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = \begin{cases} -\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right) & \text{if } y^{(i)} = 0 \end{cases}$$

$$\Rightarrow \quad L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = -y^{(i)}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) - (1 - y^{(i)})\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)$$

- If $y^{(i)} = 1$

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = -1 \times \log(f_{\vec{w},b}(\vec{x}^{(i)})) - (1-1) \times ☺ = -\log(f)$$

- If $y^{(i)} = 0$

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = -0 \times ☺ - (1-0) \times \log(1 - f_{\vec{w},b}(\vec{x}^{(i)})) = -\log(1 - f)$$

Simplified cost function

➥ For one training example:

$$L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = -y^{(i)} \log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) - (1-y^{(i)}) \log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)$$

➥ Cost function [Note the minus sign in front of summation!]:

$$J(\vec{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L\left[f_{\vec{w},b}(\vec{x}^{(i)}), y^{i}\right]$$

$$= -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) + (1-y^{(i)}) \log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)\right]$$

maximum likelihood

(don't worry about it – it's just a name from statistics ☺☺)

Simplified cost function

- For one training example:

$$L\left[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{(i)}\right] = -y^{(i)} \log\left(f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) - (1 - y^{(i)}) \log\left(1 - f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right)$$

- Cost function [Note the minus sign in front of summation!]:

$$J(\overrightarrow{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L\left[f_{\overrightarrow{w},b}(\vec{x}^{(i)}), y^{i}\right]$$

$$= -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log\left(f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - f_{\overrightarrow{w},b}(\vec{x}^{(i)})\right)\right]$$

maximum likelihood

(don't worry about it – it's just a name from statistics ☺☺)

Find $\overrightarrow{w}, b$: Minminize $J(\overrightarrow{w}, b)$ with respect to $\overrightarrow{w}, b \quad \rightarrow \quad \min_{\overrightarrow{w},b} J(\overrightarrow{w}, b)$

Convex cost function

$$J(\vec{w}, b) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log\left(f_{\vec{w},b}(\vec{x}^{(i)}) \right) + (1 - y^{(i)}) \log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)}) \right) \right]$$

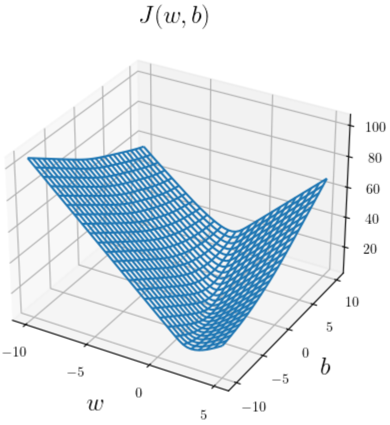


Table of Contents

Training logistic regression: Gradient descent

$$J(\vec{w}, b) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log\left(f_{\vec{w},b}(\vec{x}^{(i)}) \right) + (1 - y^{(i)}) \log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)}) \right) \right]$$

Training logistic regression: Gradient descent

$$J(\vec{w}, b) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right) \right]$$

Repeat

$$\begin{cases} w_j = w_j - \alpha \dfrac{\partial J}{\partial w_j}(\vec{w}, b) \\ \\ b = b - \alpha \dfrac{\partial J}{\partial b}(\vec{w}, b) \end{cases}$$

simultaneous updates

Training logistic regression: Gradient descent

$$J(\vec{w}, b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-f_{\vec{w},b}(\vec{x}^{(i)})\right)\right]$$

Repeat

$$\begin{cases} w_j = w_j - \alpha\dfrac{\partial J}{\partial w_j}(\vec{w}, b) \\ \\ b = b - \alpha\dfrac{\partial J}{\partial b}(\vec{w}, b) \end{cases}$$

simultaneous updates

$$\frac{\partial J}{\partial w_j}(\vec{w}, b) = \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}$$

$$\frac{\partial J}{\partial b}(\vec{w}, b) = \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(x^{(i)}) - y^{(i)}\right)1$$

looks like linear regression!

$\rightarrow$ Not the same though as $f_{\vec{w},b}$ is different

same same but different

Gradient descent for logistic regression

Repeat

$$\begin{cases} w_j = w_j - \alpha\left[\dfrac{1}{m}\displaystyle\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}\right] \\ b = b - \alpha\left[\dfrac{1}{m}\displaystyle\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)\right] \end{cases}$$

simultaneous updates

Linear regression: $f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b$

Logistic regression:

$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1 + \mathrm{e}^{-(\vec{w}\cdot\vec{x}+b)}}$$

Same concepts:

- Monitor gradient descent (learning curve)
- Vectorized implementation
- Feature scaling

LogisticRegression from sklearn

```
class sklearn.linear_model.LogisticRegression(penalty='l2', *, dual=False, tol=0.0001,
C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None,
solver='lbfgs', max_iter=100, multi_class='deprecated', verbose=0, warm_start=False,
n_jobs=None, l1_ratio=None)
```
[source]

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross-entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag', 'saga' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag', 'saga' and 'lbfgs' solvers. **Note that regularization is applied by default**. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.

Figure: LogisticRegression from sklearn

https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LogisticRegression.html

LogisticRegression from sklearn

Examples

```
>>> from sklearn.datasets import load_iris
>>> from sklearn.linear_model import LogisticRegression
>>> X, y = load_iris(return_X_y=True)
>>> clf = LogisticRegression(random_state=0).fit(X, y)
>>> clf.predict(X[:2, :])
array([0, 0])
>>> clf.predict_proba(X[:2, :])
array([[9.8...e-01, 1.8...e-02, 1.4...e-08],
       [9.7...e-01, 2.8...e-02, ...e-08]])
>>> clf.score(X, y)
0.97...
```

Figure: LogisticRegression from sklearn

https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LogisticRegression.html