Gradient descent for Logistic Regression

Gradient Descent for Logistic Regression

Given the cost function (J) and the gradient descent algorithms for w and b,

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

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

⇒ Simultaneous update

The derivative can be calculated as:

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

Then, replacing to the formulas of w_j and b:

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

Although the update rules resemble those of linear regression, the key difference is in the function $f_{\vec{w},b}(\vec{x})$. In logistic regression, it uses the **sigmoid** applied to $\vec{w} \cdot \vec{x} + b$, making it fundamentally different despite the similar form.

- Linear Regression: $f_{\vec{w},b}(\vec{x}) = \vec{w} \cdot \vec{x} + b$
- Logistic Regression: $f_{\vec{w},b}(\vec{x}) = g(\vec{w} \cdot \vec{x} + b) = \frac{1}{1+e^{-(\vec{w} \cdot \vec{x} + b)}}$

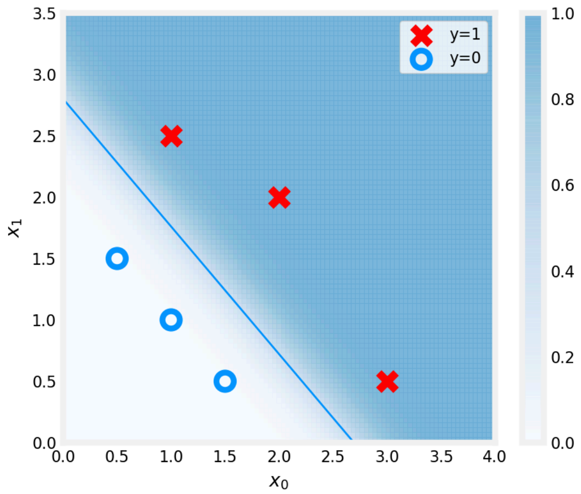
💡 **Same concept:**

- Monitoring Gradient Descent. *(learning curve)*
- Vectorized Implementation. *(to improve the efficiency of gradient descent)*
- Feature Scaling. *(enhance the convergence speed of gradient descent)*

Example:

Running Gradient Descent and the cost decreases for each loop.

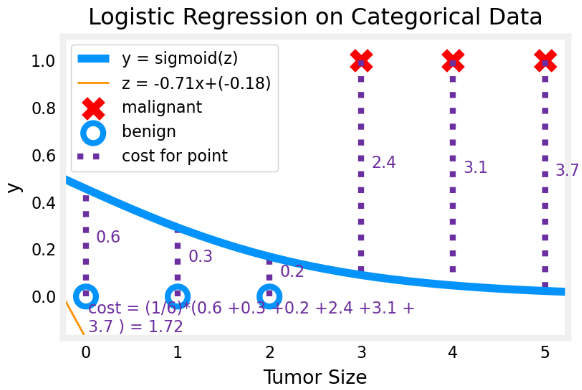
```
Iteration    0: Cost 0.684610468560574
Iteration 1000: Cost 0.1590977666870456
Iteration 2000: Cost 0.08460064176930081
Iteration 3000: Cost 0.05705327279402531
Iteration 4000: Cost 0.042907594216820076
Iteration 5000: Cost 0.034338477298845684
Iteration 6000: Cost 0.028603798022120097
Iteration 7000: Cost 0.024501569608793
Iteration 8000: Cost 0.02142370332569295
Iteration 9000: Cost 0.019030137124109114

updated parameters: w:[5.28 5.08], b:-14.222409982019837
```

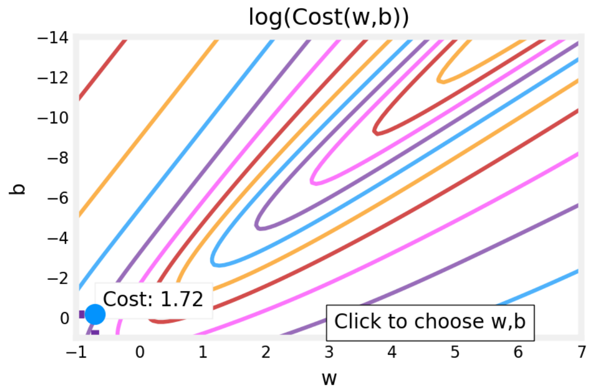


In the following plot:

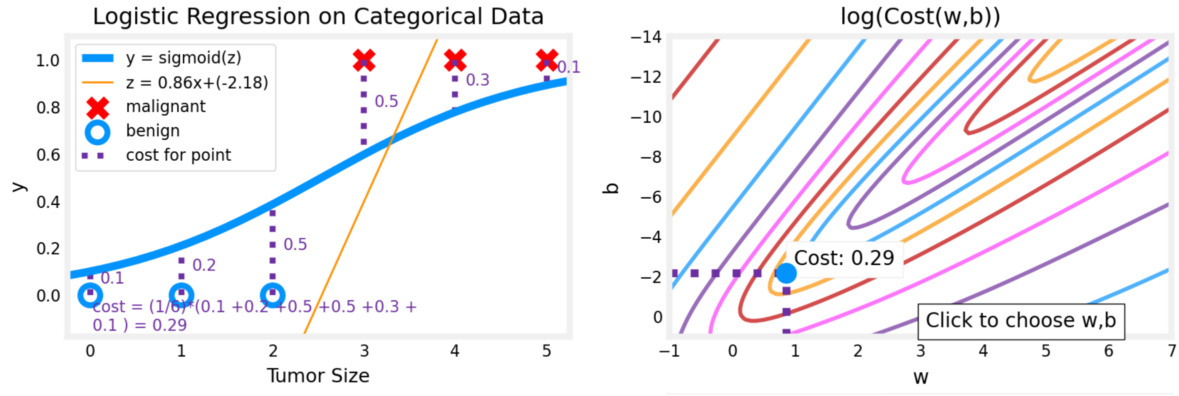
- the shading reflects the probability $y = 1$ (result prior to decision boundary)
- the decision boundary is the line at which the probability =0.5

For this postion of w and b ⇒ the Sigmoid shape is not good

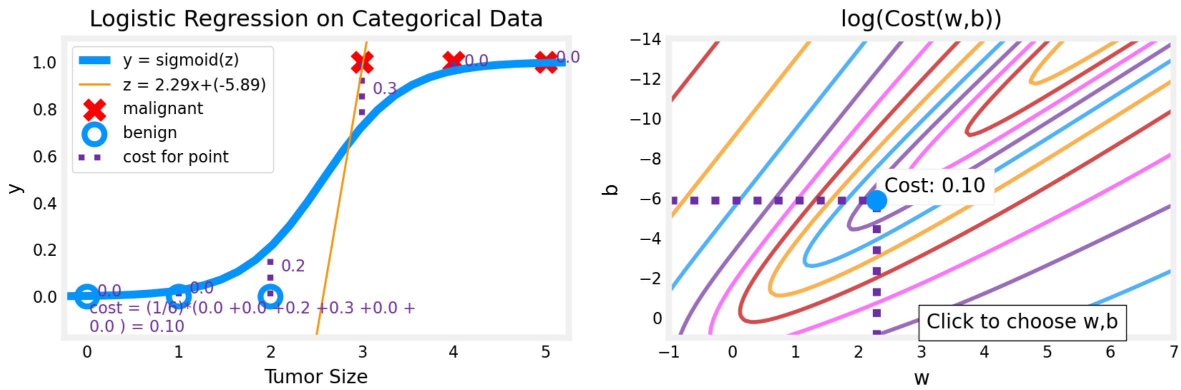




It is a little better



Better:



It is the best cost for logistic regression.