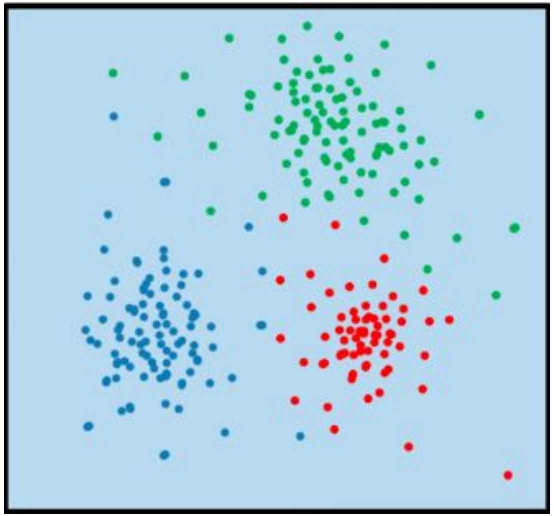
Machine Learning

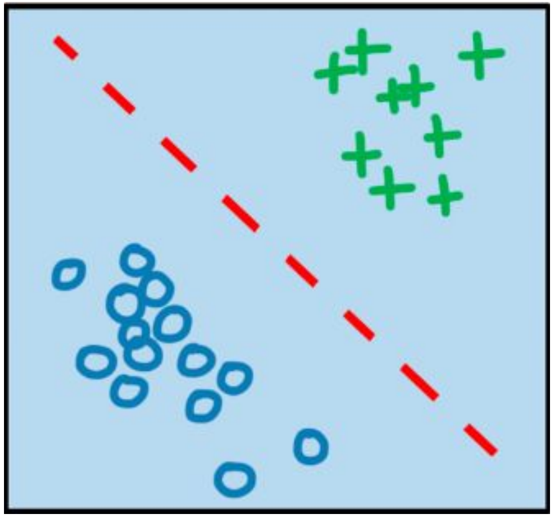
Lesson 2

Machine Learning

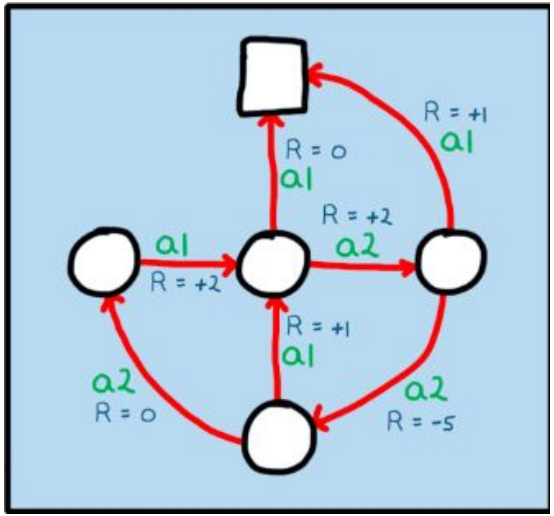
Unsupervised



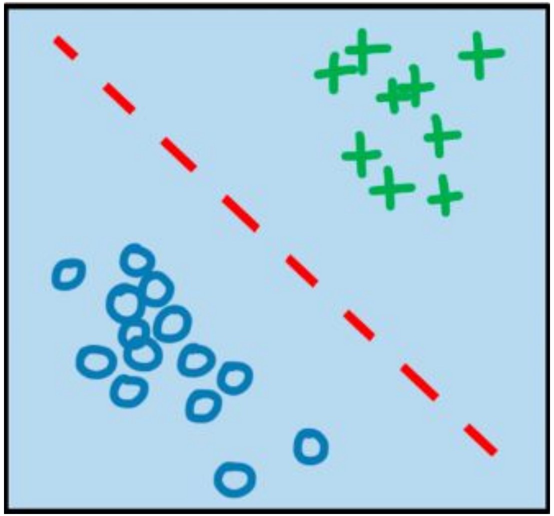
Supervised



Reinforcement



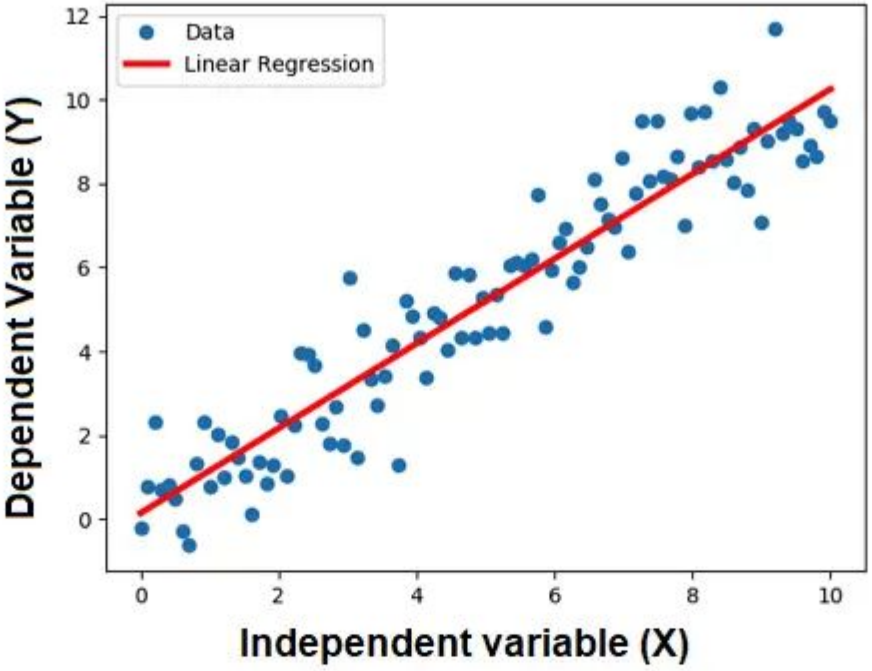
Supervised Learning



Regression

Classification

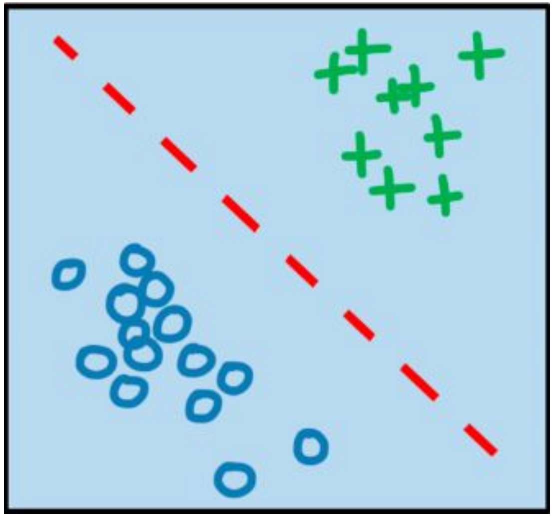
Supervised Learning



X	Y
1	2
3	4
3	5
6	7
7	7
3	4

. .
. .
. .

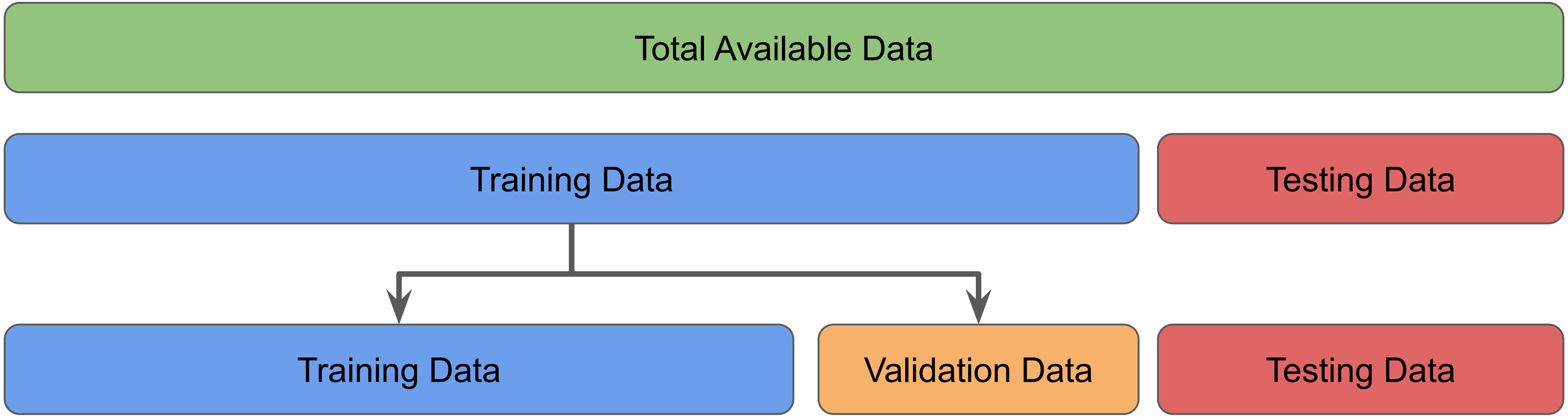
Supervised Learning



x1	x2
1	2
3	4
3	2
6	3
7	4
3	1
.	.
.	.
.	.

Label
O
O
X
X
X
O
.
.
.

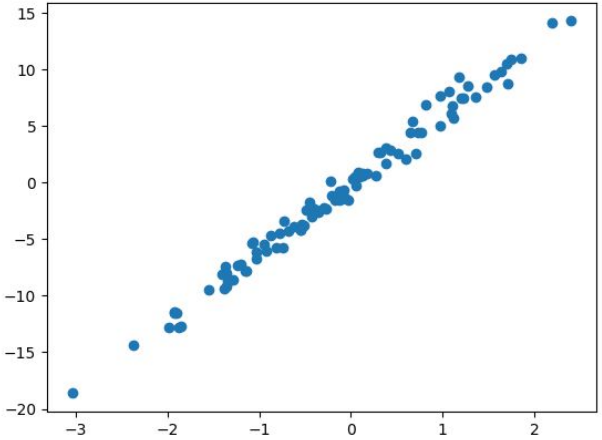
Dataset

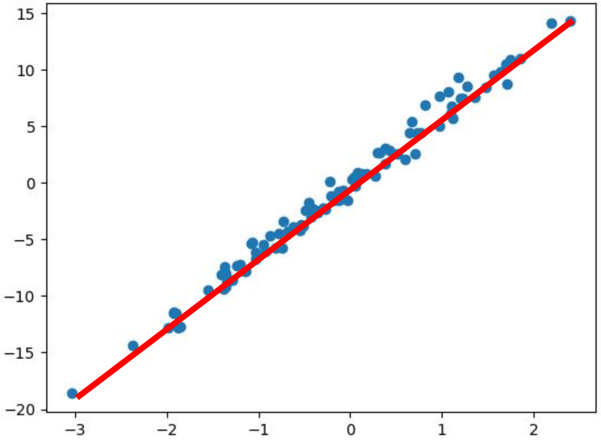

Total Available Data
Training Data
Testing Data
Training Data
Validation Data
Testing Data

Supervised Machine Learning Algorithms

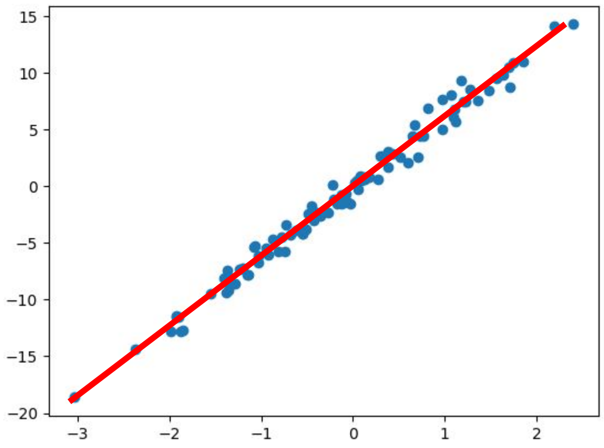
- Nearest Neighbor
- Naive Bayes
- Decision Trees
- Linear Regression
- Logistic Regression
- Support Vector Machines (SVM)
- Neural Networks

Linear Regression





Linear Regression



$$y = \theta x + b$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y_i} - y)^2$$

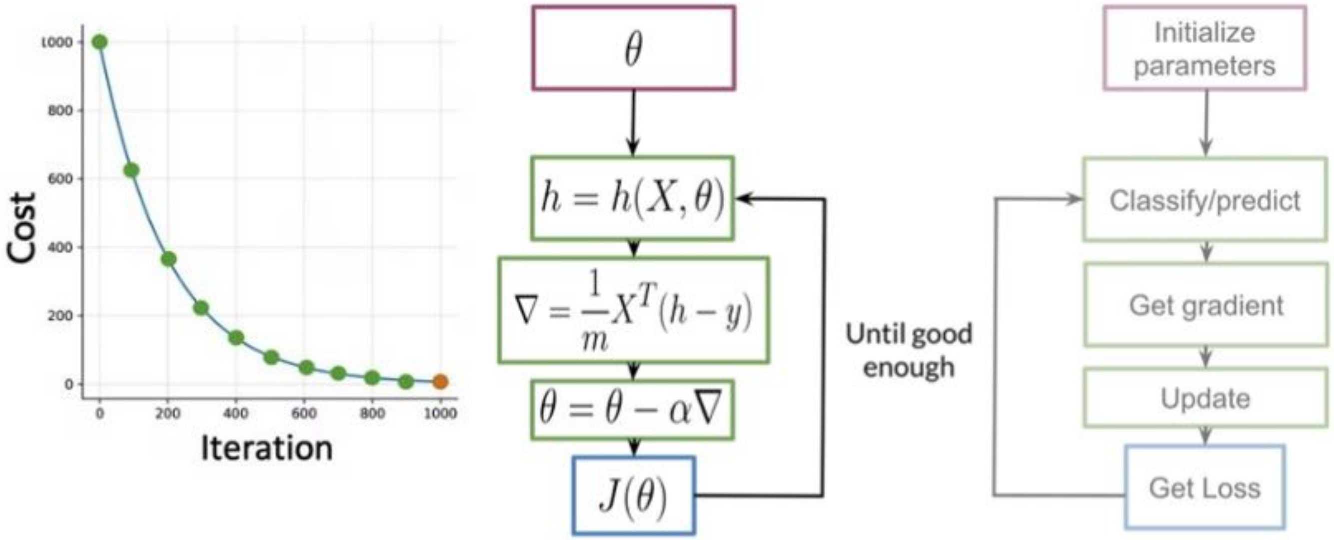
$$L(\theta, b) = \frac{1}{N}((x\theta + b) - y)^2$$

$$\frac{\mathrm{d}L}{\mathrm{d}\theta} = \frac{1}{N}x^T((x\theta + b) - y)$$

$$\frac{\mathrm{d}L}{\mathrm{d}b} = \frac{1}{N}((x\theta + b) - y)$$

Linear Regression: Training

Usually you keep training until the cost converges. If you were to plot the number of iterations versus the cost, you should see something like this:



You initialize your parameter **θ**, that you can use in your equation, you then compute the gradient that you will use to update **θ**, and then calculate the cost. You keep doing so until good enough.

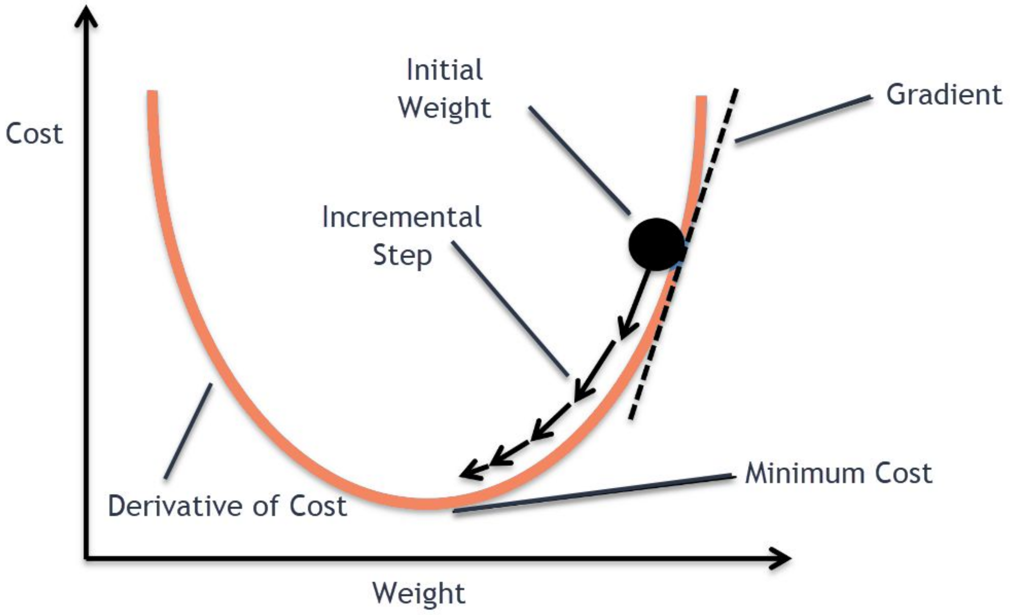
Linear Regression : Gradient Descent

$$\min_{\theta} J(\theta)$$

$$\theta \leftarrow \theta - \alpha \frac{\mathrm{d}}{\mathrm{d}\theta} J(\theta)$$

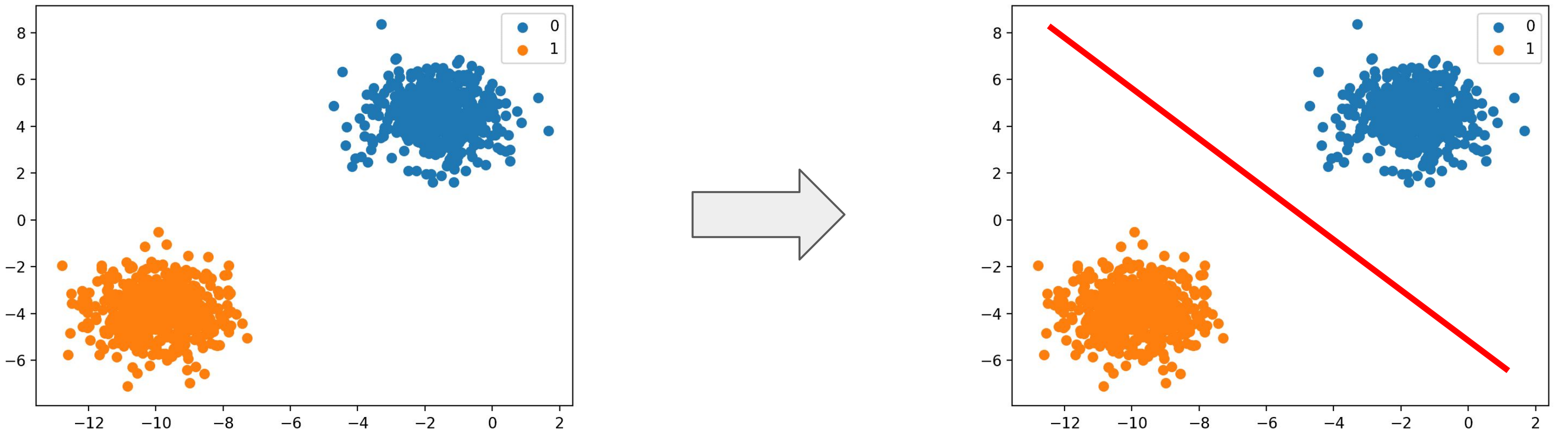
$$\frac{\mathrm{d}}{\mathrm{d}\theta} J(\theta) = (h(x, \theta) - y)\, x$$

$$\theta \leftarrow \theta - \alpha(h(x, \theta) - y)x$$

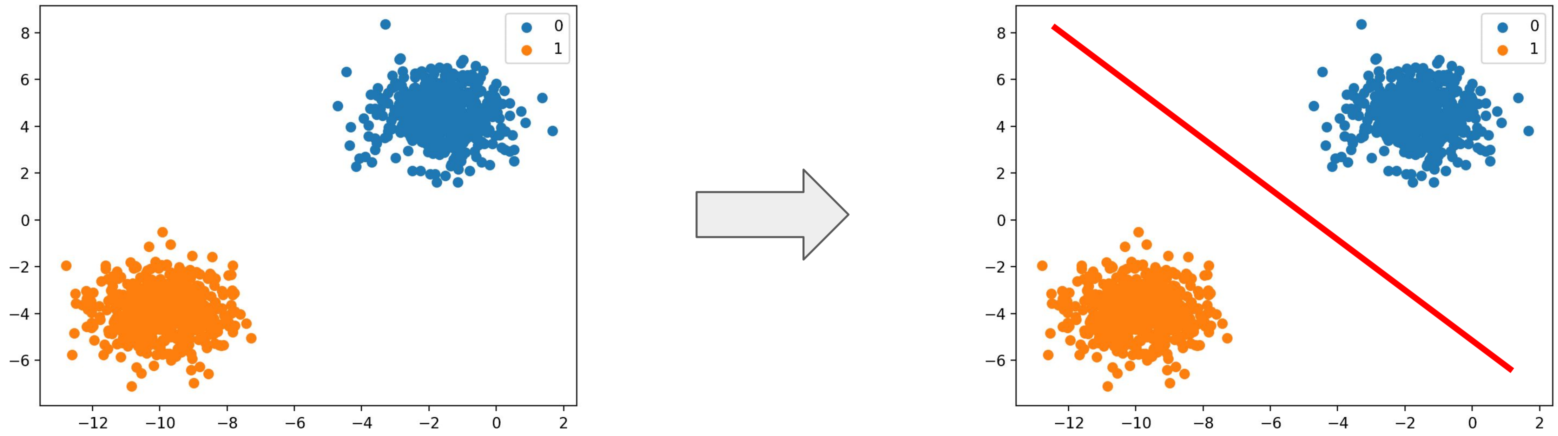


α → learning rate

Logistic Regression



Logistic Regression



$$x_2 = \theta . x_1 + b$$

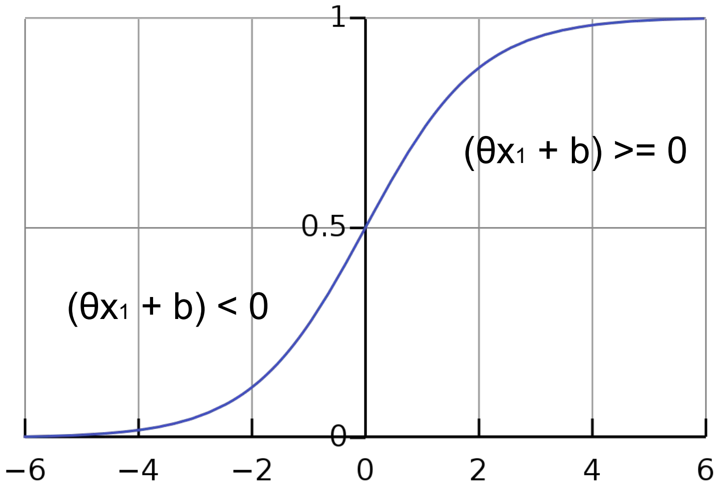
Logistic Regression

But we don't need $\mathbf{x_2}$, we need probability that a point is on either side of that line.

So we use the sigmoid function,

$$f(x) = \frac{1}{1+e^{-x}}$$

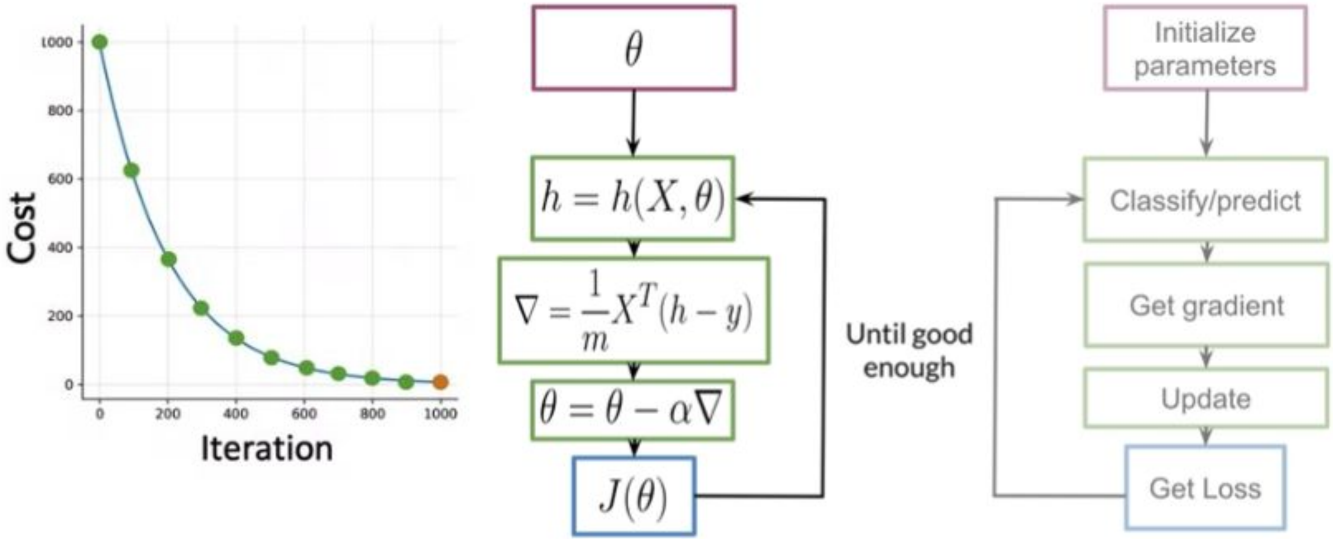
$$P = f(\theta x_1 + b) = \frac{1}{1 + e^{-(\theta x_1 + b)}}$$



Note that as $(\theta x_1 + b)$ gets closer and closer to $-\infty$ the denominator of the sigmoid function gets larger and larger and as a result, the sigmoid gets closer to 0. On the other hand, as $(\theta x_1 + b)$ gets closer and closer to $+\infty$ the denominator of the sigmoid function gets closer to 1 and as a result the sigmoid also gets closer to 1.

Logistic Regression : Training

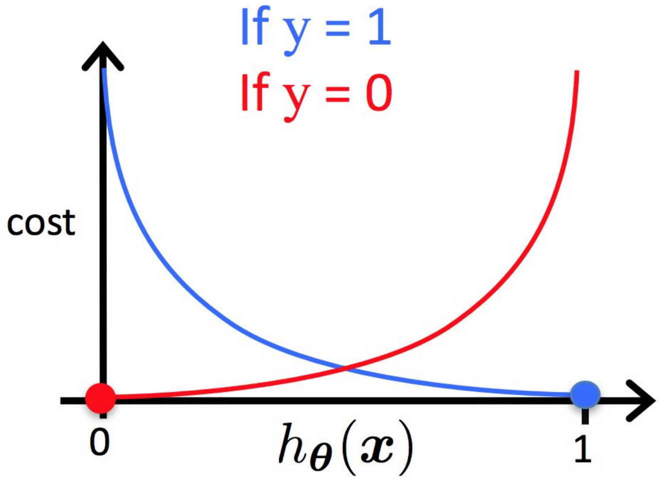
Usually you keep training until the cost converges. If you were to plot the number of iterations versus the cost, you should see something like this:



You initialize your parameter **θ**, that you can use in your sigmoid, you then compute the gradient that you will use to update **θ**, and then calculate the cost. You keep doing so until good enough.

Logistic Regression : Cost Function

$$J(\theta) = -\left[y \log h(x, \theta) + (1 - y) \log \left(1 - h(x, \theta)\right)\right]$$



As you can see in the picture above, if y = 1 and you predict something close to 0, you get a cost close to ∞. The same applies for then y=0 and you predict something close to 1. On the other hand if you get a prediction equal to the label, you get a cost of 0. In either, case you are trying to minimize **J(θ)**.

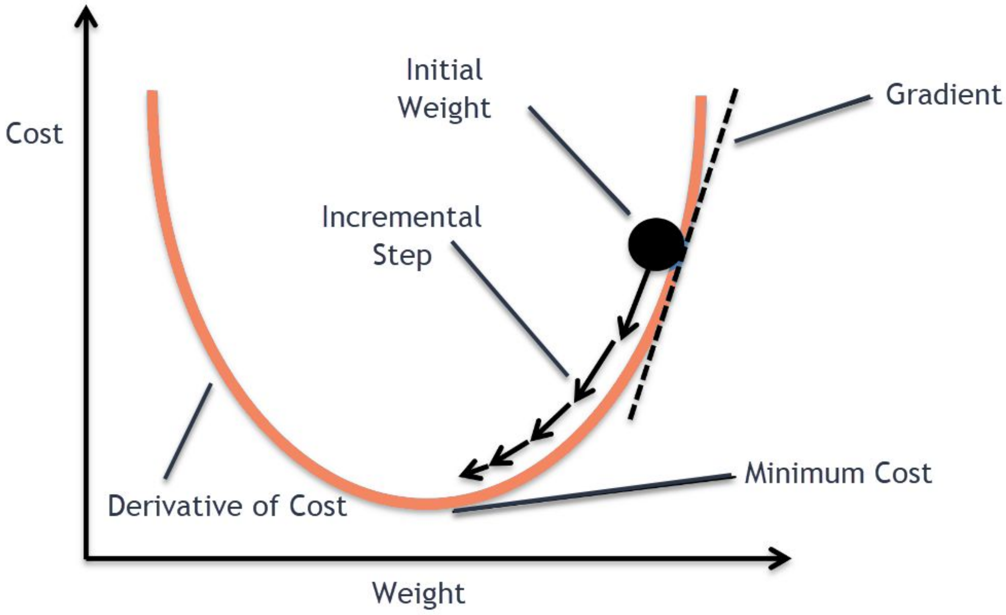
Logistic Regression : Gradient Descent

$$\min_{\theta} J(\theta)$$

$$\theta \leftarrow \theta - \alpha \frac{\mathrm{d}}{\mathrm{d}\theta} J(\theta)$$

$$\frac{\mathrm{d}}{\mathrm{d}\theta} J(\theta) = (h(x, \theta) - y)\, x$$

$$\theta \leftarrow \theta - \alpha(h(x, \theta) - y)x$$



α → learning rate