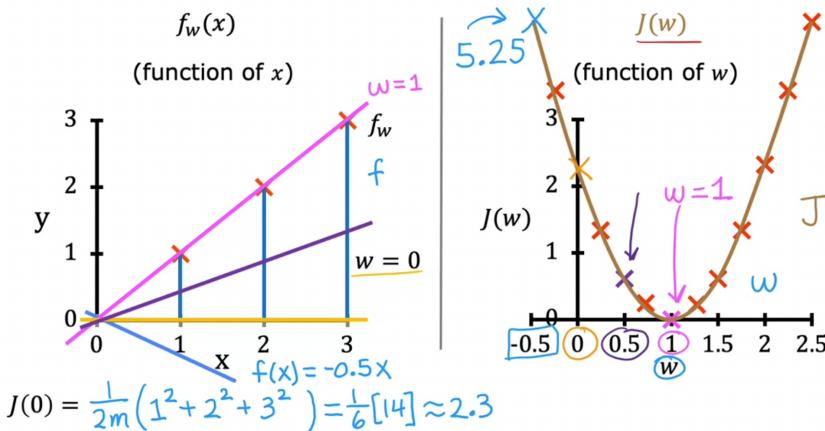


Supervised Machine Learning: Regression and Classification

Week 1 - Introduction to Machine Learning

This week they talked about supervised and unsupervised learning, regression models and gradient descent.

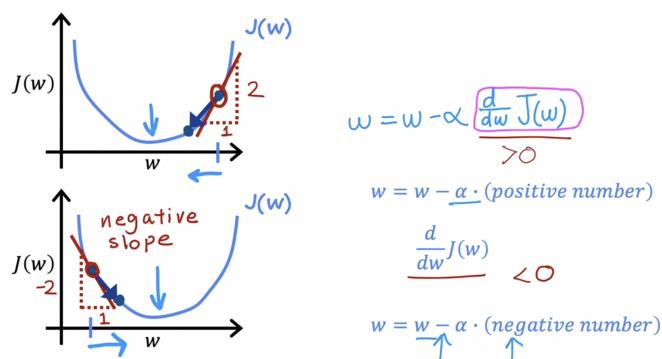
Below you can see some intuition for the **cost function**, different parameters of w resulting in different costs.



Now you can see the implementation of the gradient descent algorithm.

Gradient descent algorithm		Assignment	Truth assertion
Repeat until convergence		$\alpha = c$	$a = c$
$w = w - \alpha \frac{\partial}{\partial w} J(w, b)$	Learning rate Derivative	$\alpha = \alpha + 1$	$a = a + 1$
$b = b - \alpha \frac{\partial}{\partial b} J(w, b)$	Simultaneously update w and b	Code	Math $a == c$
Correct: Simultaneous update		Incorrect	
$tmp_w = w - \alpha \frac{\partial}{\partial w} J(w, b)$		$tmp_w = w - \alpha \frac{\partial}{\partial w} J(w, b)$	
$tmp_b = b - \alpha \frac{\partial}{\partial b} J(w, b)$		$w = tmp_w$	
$w = tmp_w$		$tmp_b = b - \alpha \frac{\partial}{\partial b} J(w, b)$	
$b = tmp_b$		$b = tmp_b$	

Then you can see the intuition for gradient descent below.



Week 2 - Regression with multiple input variables

This week they talked about multiple linear regression and gradient descent in practice.

One example of a model with multiple variables is shown below.

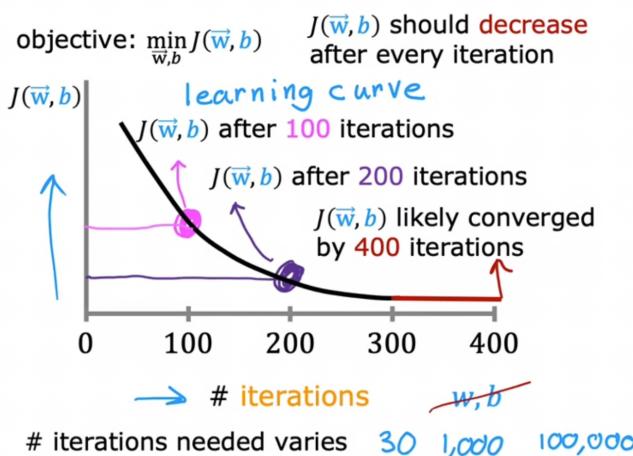
$$\begin{aligned}
 f_{\vec{w}, b}(\vec{x}) &= w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b \\
 \vec{w} &= [w_1 \ w_2 \ w_3 \dots \ w_n] \quad \text{parameters of the model} \\
 b &\text{ is a number} \\
 \vec{x} &= [x_1 \ x_2 \ x_3 \dots \ x_n] \\
 f_{\vec{w}, b}(\vec{x}) &= \vec{w} \cdot \vec{x} + b = w_1 x_1 + w_2 x_2 + w_3 x_3 + \dots + w_n x_n + b \\
 &\quad \uparrow \text{dot product}
 \end{aligned}$$

The gradient descent algorithm for multiple variables look like this:

Gradient descent	
<p>repeat {</p> <p style="margin-left: 20px;">One feature</p> $\vec{w}_j = \vec{w} - \alpha \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) \underline{x_j^{(i)}}$ <p style="margin-left: 20px;">$\hookrightarrow \frac{\partial}{\partial w_j} J(\vec{w}, b)$</p> $b = b - \alpha \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)})$ <p style="margin-left: 20px;">simultaneously update w, b</p> <p>}</p>	<p>repeat {</p> <p style="margin-left: 20px;">n features ($n \geq 2$)</p> $\vec{w}_1 = \vec{w}_1 - \alpha \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) \underline{x_1^{(i)}}$ <p style="margin-left: 20px;">\vdots</p> $\vec{w}_n = \vec{w}_n - \alpha \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) \underline{x_n^{(i)}}$ $b = b - \alpha \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)})$ <p style="margin-left: 20px;">simultaneously update w_j (for $j = 1, \dots, n$) and b</p> <p>}</p>

Then, for checking the convergence of the algorithm you should look at the decrease of the cost per iteration.

Make sure gradient descent is working correctly



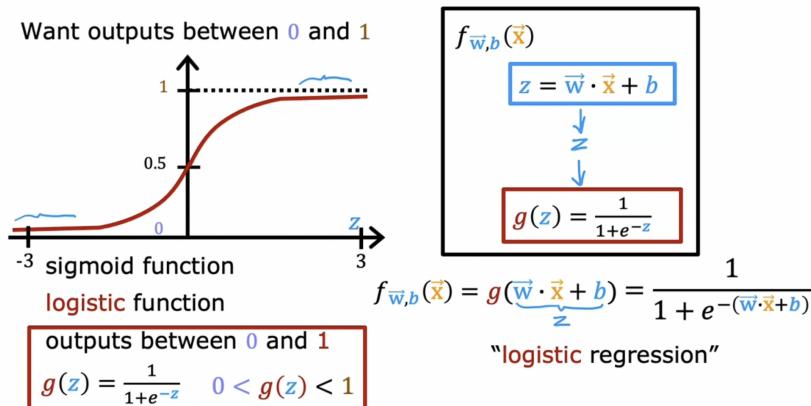
Automatic convergence test
Let ϵ "epsilon" be 10^{-3} .
 0.001

If $J(\vec{w}, b)$ decreases by $\leq \epsilon$ in one iteration,
declare convergence.
(found parameters \vec{w}, b to get close to global minimum)

Week 3 - Classification

This week they talked about classification with logistic regression, cost function, and gradient descent, and also touched base on the problem of overfitting.

The main difference between linear and logistic regression is the link function.



Below you can see the cost function for the logistic regression.

Cost

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

cost

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

loss

Convex
↳ can reach a global minimum

The gradient descent for logistic regression is presented below.

Gradient descent for logistic regression

```
repeat {
    looks like linear regression!
     $w_j = w_j - \alpha \left[ \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) x_j^{(i)} \right]$ 
     $b = b - \alpha \left[ \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) \right]$ 
} simultaneous updates
```

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

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

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