Gradient Descent: The Code

From before we saw that one weight update can be calculated as:

$$\Delta w_i = \eta \, \delta x_i$$

with the error term δ as

$$\delta = (y - \hat{y}) f'(h) = (y - \hat{y}) f'(\sum w_i x_i)$$

Remember, in the above equation $(y - \hat{y})$ is the output error, and $f'(h)$ refers to the derivative of the activation function, $f(h)$. We'll call that derivative the output gradient.

Now I'll write this out in code for the case of only one output unit. We'll also be using the sigmoid as the activation function $f(h)$.

```
# Defining the sigmoid function for activations
def sigmoid(x):
    return 1/(1+np.exp(-x))

# Derivative of the sigmoid function
def sigmoid_prime(x):
    return sigmoid(x) * (1 - sigmoid(x))

# Input data
x = np.array([0.1, 0.3])
# Target
y = 0.2
# Input to output weights
weights = np.array([-0.8, 0.5])

# The learning rate, eta in the weight step equation
learnrate = 0.5

# the linear combination performed by the node (h in f(h) and f'(h))
h = x[0]*weights[0] + x[1]*weights[1]
# or h = np.dot(x, weights)

# The neural network output (y-hat)
nn_output = sigmoid(h)

# output error (y - y-hat)
error = y - nn_output

# output gradient (f'(h))
output_grad = sigmoid_prime(h)

# error term (lowercase delta)
error_term = error * output_grad

# Gradient descent step
del_w = [ learnrate * error_term * x[0],
          learnrate * error_term * x[1]]
# or del_w = learnrate * error_term * x
```

gradient.py | solution.py

```
1  import numpy as np
2
3  def sigmoid(x):
4      """
5      Calculate sigmoid
6      """
7      return 1/(1+np.exp(-x))
8
9  def sigmoid_prime(x):
10     """
```

```
learnrate = 0.5
x = np.array([1, 2, 3, 4])
y = np.array(0.5)

# Initial weights
w = np.array([0.5, -0.5, 0.3, 0.1])

### Calculate one gradient descent step for each weight
### Note: Some steps have been consilated, so there are
###       fewer variable names than in the above sample code

# TODO: Calculate the node's linear combination of inputs and weights
h = 0
for xi, wi in zip(x,w):
    h += xi * wi

# TODO: Calculate output of neural network
```

RESET QUIZ | TEST RUN | SUBMIT ANSWER

NEXT