Project-2: Documentation

Anudeep Reddy

10/17/18

1 Perceptron

In 'perceptron_train()', I used final_w & final_b to store final weights & bias and are initialized to zeros. Activation is calculated while iterating through each sample and lable simultaneously and stored in 'a'. If y*a $\leq$ 0, weights are updated and stored in final_w and final_b. Weights, bias & sample number at which update was done are stored in seperate variables namely temp_w, temp_b & x_change_in_nxt_epoch which are used to check in the next epoch at the same sample number if the weights & bias are equal. If equal loop doesn't run another epoch. And final_w, final_b are returned as list.

In perceptron_test(), variable 'accuarcy' is first set to zero and with each sample, activation 'a' is calculated and accuarcy is increased by 1 if y*a $>$ 0 is satisfied. Finally, accuracy is divided by total number of samples and multiplied by 100 and returned.

After training with the data, weights = [2, 4] and bias = 2.

2 Gradient Descent

In gradient_descent(), X is updated using $X = X - (eta * dev(X))$. Gradient values of X are calculated and values less than 0.0001 are not updated next time. So, only gradient values greater than 0.0001 are updated every time. Finally, if all the gradient values are less than the threshold, loop is stopped and final X is returned.

When $f(x) = x_1^2 + x_2^2$, $x_{init} = (5.0, 5.0)$ and $\eta = 0.1$.

$\nabla f = [2 * x_1, 2 * x_2]$

We get, X $= [4.56719262e - 05, 4.56719262e - 05]$