Early Stop

Let n be the number of steps between evaluations.
Let p be the "patience," the number of times to observe worsening validation set error before giving up.
Let $\boldsymbol{\theta}_o$ be the initial parameters.

Initialization

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_o$$
$$i \leftarrow 0$$
$$j \leftarrow 0$$
$$v \leftarrow \infty$$
$$\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}$$
$$i^* \leftarrow i$$

θ is the vector of weights

i, j are the loop index

Validation Error

Algorithm

while $j < p$ **do**
  Update $\boldsymbol{\theta}$ by running the training algorithm for n steps.
  $i \leftarrow i + n$
  $v' \leftarrow \text{ValidationSetError}(\boldsymbol{\theta})$
  if $v' < v$ **then**
    $j \leftarrow 0$
    $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}$
    $i^* \leftarrow i$
    $v \leftarrow v'$
  else
    $j \leftarrow j + 1$
  end if
end while
Best parameters are $\boldsymbol{\theta}^*$, best number of training steps is i^*

Gradient Descent

Requirements

- Choose Basis Function:
  - How to choose the basis functions?
  - What do your basis functions look like?
  - The number of basis functions
- Closed Form Solution: give me some immediate results

- Gradient Descent:
  - How to compute the Gradient
  - Learning rate
  - Early Stop Parameters

Requirements

- Evaluation:

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^*)/N_{\mathrm{V}}}$$

- Coding:
  - Organized: functionalities should be realized by functions (or classes) respectively
  - Clear Comments: Explain input and output for all the functions
  - No Redundant Comments
- Points off if your code is badly organized, even your result is correct. (Cuz we can't test all the cases, there must be bugs in a bad code)