An incremental linear perceptron based on the BFGS algorithm

Flávio Eler De Melo

April 5, 2019

Abstract

This document briefly describes a new procedure for linear incremental learning that minimises the least-square residues recursively based on the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm.

1 Least squares regressor

Let X and Y be the input and output matrices with dimensions $n_x \times m$ and $n_y \times m$ respectively, where m is the number of examples. The least squares problem requires to find the weight matrix W (of dimension $n_x \times n_y$) that minimizes the loss:

$$\mathcal{L}(\mathrm{X}, \mathrm{Y}, \mathrm{W}) = \frac{1}{2} \sum_{j=1}^{n_y} \sum_{k=1}^{m} \left(y_{jk} - \sum_{\ell=1}^{n_x} w_{\ell j} x_{\ell k} \right)^2 . \tag{1.1}$$

Note that:

$$\begin{aligned}
\nabla_{\mathrm{W}} \mathcal{L}(\mathrm{X}, \mathrm{Y}, \mathrm{W}) = \left[\nabla_{\mathrm{W}} \mathcal{L}(\mathrm{X}, \mathrm{Y}, \mathrm{W})\right]_{ij} &= -\sum_{k=1}^{m} x_{ik} \left(y_{jk} - \sum_{\ell=1}^{n_x} w_{\ell j} x_{\ell k} \right) \\
&= -\mathrm{X} \left(\mathrm{Y}^{\mathrm{T}} - \mathrm{X}^{\mathrm{T}} \mathrm{W} \right) \equiv \mathbf{0}_{n_x \times n_y} . \\
\mathrm{X}\mathrm{X}^{\mathrm{T}} \mathrm{W} &= \mathrm{X}\mathrm{Y}^{\mathrm{T}}, \\
\mathrm{H}\mathrm{W} &= \mathrm{X}\mathrm{Y}^{\mathrm{T}}, \\
\mathrm{W} &= \mathrm{H}^{-1} \mathrm{X}\mathrm{Y}^{\mathrm{T}},
\end{aligned}$$

where $\mathrm{H} = \mathrm{H}^{\mathrm{T}} = \mathrm{X}\mathrm{X}^{\mathrm{T}}$ (symmetric). Therefore, a(n) (initial) solution can be calculated for the first collected examples $(\mathrm{X}_0, \mathrm{Y}_0)$ by:

$$\mathrm{W}_0 = \mathrm{H}_0^{-1} \mathrm{X}_0 \mathrm{Y}_0^{\mathrm{T}}, \tag{1.2}$$

where $\mathrm{H}_0 = \mathrm{X}_0 \mathrm{X}_0^{\mathrm{T}}$, $\mathrm{X}_0 = [x_{0,ij}]_{i \in [1..n_x], j \in [1..m]}$ at step $k = 0$ for n_x dimensions (including the constant term) of the independent variable vector and m training examples, $\mathrm{Y}_0 = [y_{0,ij}]_{i \in [1..n_y], j \in [1..m]}$ at step $k = 0$ for n_y dimensions of the dependent variable vector.

2 Incremental update

Now suppose that we will search for an incremental minimisation of $\mathcal{L}(\mathrm{X}_k, \mathrm{Y}_k, \mathrm{W}_k)$. This can be done by finding the direction p_k of minimisation via the (quasi) Newton equation:

$$\mathrm{H}_k \mathrm{p}_k = -\nabla_{\mathrm{W}} \mathcal{L}(\mathrm{X}_k, \mathrm{Y}_k, \mathrm{W}_k) = \mathrm{X}_k \left(\mathrm{Y}_k^{\mathrm{T}} - \mathrm{X}_k^{\mathrm{T}} \mathrm{W}_k \right). \tag{2.1}$$

Note that for $\mathrm{H}_0 = \mathrm{X}_0 \mathrm{X}_0^{\mathrm{T}}$,

$$\begin{aligned} \mathrm{p}_0 &= \mathrm{H}_0^{-1} \mathrm{X}_0 \left(\mathrm{Y}_0^{\mathrm{T}} - \mathrm{X}_0^{\mathrm{T}} \mathrm{W}_0 \right) \\ &= \mathrm{H}_0^{-1} \mathrm{X}_0 \mathrm{Y}_0^{\mathrm{T}} - \mathrm{H}_0^{-1} \mathrm{X}_0 \mathrm{X}_0^{\mathrm{T}} \mathrm{W}_0 \\ &= \mathrm{W}_0 - \mathrm{W}_0 = \mathbf{0}, \end{aligned}$$

i.e., the loss functional is already at a minimum. By assumption, H_{k-1}^{-1} and W_{k-1} are parameters kept to induce the incremental regression. Therefore, given new examples $(\mathrm{X}_k, \mathrm{Y}_k)$, our incremental task involves inducing the procedure by finding the incremental direction via

$$\mathrm{p}_k = -\mathrm{H}_{k-1}^{-1} \nabla_{\mathrm{W}} \mathcal{L}(\mathrm{X}_k, \mathrm{Y}_k, \mathrm{W}_{k-1}) = \mathrm{H}_{k-1}^{-1} \mathrm{X}_k \left(\mathrm{Y}_k^{\mathrm{T}} - \mathrm{X}_k^{\mathrm{T}} \mathrm{W}_{k-1} \right). \tag{2.2}$$

In the context of the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm, the problem requires line search to find a proper incremental factor as:

$$\boldsymbol{\alpha}_k = \mathrm{argmin}_{\boldsymbol{\alpha}} \mathcal{L}(\mathrm{X}_k, \mathrm{Y}_k, \mathrm{W}_k(\boldsymbol{\alpha})), \tag{2.3}$$

where $\mathrm{W}_k(\boldsymbol{\alpha}) = \mathrm{W}_{k-1} + \boldsymbol{\alpha} \mathrm{p}_k$, which can be computed analytically by

$$\begin{aligned} \nabla_{\boldsymbol{\alpha}} \mathcal{L}(\mathrm{X}_k, \mathrm{Y}_k, \mathrm{W}_k(\boldsymbol{\alpha})) &= \nabla_{\mathrm{W}}^{\mathrm{T}} \mathcal{L}(\mathrm{X}_k, \mathrm{Y}_k, \mathrm{W}_k) \nabla_{\boldsymbol{\alpha}} \mathrm{W}_k(\boldsymbol{\alpha}) \\ &= -\left(\mathrm{Y}_k - \mathrm{W}_k^{\mathrm{T}} \mathrm{X}_k \right) \mathrm{X}_k^{\mathrm{T}} \mathrm{p}_k \equiv \mathbf{0}_{n_y \times n_y}, \\ \boldsymbol{\alpha}_k \mathrm{p}_k^{\mathrm{T}} \mathrm{X}_k \mathrm{X}_k^{\mathrm{T}} \mathrm{p}_k &= \left(\mathrm{Y}_k - \mathrm{W}_{k-1}^{\mathrm{T}} \mathrm{X}_k \right) \mathrm{X}_k^{\mathrm{T}} \mathrm{p}_k, \\ \boldsymbol{\alpha}_k &= \left(\mathrm{Y}_k - \mathrm{W}_{k-1}^{\mathrm{T}} \mathrm{X}_k \right) \mathrm{X}_k^{\mathrm{T}} \mathrm{p}_k \left(\mathrm{p}_k^{\mathrm{T}} \mathrm{X}_k \mathrm{X}_k^{\mathrm{T}} \mathrm{p}_k \right)^{-1}. \end{aligned} \tag{2.4}$$

However, instead we will define a learning rate α_k proportional to a value of reference α_{r} (a preset learning rate) that balances the effectiveness of negatives and positives for the regression. For instance, given $m = p + n$ examples, with p positives and n negatives, $\alpha_{p,k} = \alpha_{\mathrm{r}} n/m$ and $\alpha_{n,k} = \alpha_{\mathrm{r}} p/m$ so that $\alpha_{p,k} + \alpha_{n,k} = \alpha_{\mathrm{r}}$. Note that for $p > n$ the negative examples will have a higher factor to compensate for their rarer (and so weaker) contribution whereas for $n > p$ the positive examples will have a higher factor.

Now, for each example, compute the weights increment and the loss gradient increment by:

$$\delta \mathrm{W}_k = \alpha_k \mathrm{p}_k, \tag{2.5}$$

$$\begin{aligned} \mathrm{z}_k &= \nabla_{\mathrm{W}} \mathcal{L}(\mathrm{X}_k, \mathrm{Y}_k, \mathrm{W}_{k-1} + \delta \mathrm{W}_k) - \nabla_{\mathrm{W}} \mathcal{L}(\mathrm{X}_k, \mathrm{Y}_k, \mathrm{W}_{k-1}) \\ &= -\mathrm{X}_k \left(\mathrm{Y}_k^{\mathrm{T}} - \mathrm{X}_k^{\mathrm{T}} \mathrm{W}_{k-1} - \mathrm{X}_k^{\mathrm{T}} \alpha_k \mathrm{p}_k \right) + \mathrm{X}_k \left(\mathrm{Y}_k^{\mathrm{T}} - \mathrm{X}_k^{\mathrm{T}} \mathrm{W}_{k-1} \right) \\ &= \mathrm{X}_k \mathrm{X}_k^{\mathrm{T}} \alpha_k \mathrm{p}_k = \mathrm{X}_k \mathrm{X}_k^{\mathrm{T}} \delta \mathrm{W}_k. \end{aligned} \tag{2.6}$$

And finally, update the weights and the inverse Hessian matrix by the BFGS algorithm and the Sherman-Morrison formula by:

$$\mathrm{W}_k = \mathrm{W}_{k-1} + \delta \mathrm{W}_k, \tag{2.7}$$

$$\mathrm{H}_k^{-1} = \left(\mathbb{I} - \frac{\delta \mathrm{W}_k \mathrm{z}_k^{\mathrm{T}}}{\mathrm{z}_k^{\mathrm{T}} \delta \mathrm{W}_k}\right) \mathrm{H}_{k-1}^{-1} \left(\mathbb{I} - \frac{\mathrm{z}_k \delta \mathrm{W}_k^{\mathrm{T}}}{\mathrm{z}_k^{\mathrm{T}} \delta \mathrm{W}_k}\right) + \frac{\delta \mathrm{W}_k \delta \mathrm{W}_k^{\mathrm{T}}}{\mathrm{z}_k^{\mathrm{T}} \delta \mathrm{W}_k}, \tag{2.8}$$

where:

$$\delta \mathrm{W}_k \mathrm{z}_k^{\mathrm{T}} = \delta \mathrm{W}_k \left(\mathrm{X}_k \mathrm{X}_k^{\mathrm{T}} \delta \mathrm{W}_k\right)^{\mathrm{T}} = \delta \mathrm{W}_k \left(\mathrm{X}_k^{\mathrm{T}} \delta \mathrm{W}_k\right)^{\mathrm{T}} \mathrm{X}_k^{\mathrm{T}} = \delta \mathrm{W}_k \delta \mathrm{W}_k^{\mathrm{T}} \mathrm{X}_k \mathrm{X}_k^{\mathrm{T}}, \tag{2.9}$$

$$\mathrm{z}_k \delta \mathrm{W}_k^{\mathrm{T}} = \mathrm{X}_k \mathrm{X}_k^{\mathrm{T}} \delta \mathrm{W}_k \delta \mathrm{W}_k^{\mathrm{T}} = \left(\delta \mathrm{W}_k \mathrm{z}_k^{\mathrm{T}}\right)^{\mathrm{T}}, \tag{2.10}$$

$$\mathrm{z}_k^{\mathrm{T}} \delta \mathrm{W}_k = \left(\mathrm{X}_k \mathrm{X}_k^{\mathrm{T}} \delta \mathrm{W}_k\right)^{\mathrm{T}} \delta \mathrm{W}_k = \delta \mathrm{W}_k^{\mathrm{T}} \mathrm{X}_k \mathrm{X}_k^{\mathrm{T}} \delta \mathrm{W}_k. \tag{2.11}$$

The procedure is repeated until all examples have been taken into account.