Published in Image Processing On Line on YYYY–MM–DD.
Submitted on YYYY–MM–DD, accepted on YYYY–MM–DD.
ISSN 2105–1232 © YYYY IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://ipolcore.ipol.im/demo/clientApp/demo.html?id=77777000527

Double Descent in the case of Multilayer Perceptron

Emett Haddad

ENS Paris-Saclay, 91190, Gif-Sur-Yvette, France
emett.haddad@ens-paris-saclay.fr

PREPRINT June 11, 2024

Abstract

We consider $y^\star : x \in \mathcal{X} = \mathbb{R}^D \to \mathbb{E}_{(x,y)\sim\mathcal{P}}(y|x) \in \mathcal{Y} = \mathbb{R}$, with x and y random variable with joint distribution $(x, y) \hookrightarrow \mathcal{P}$. We have a **dataset** $\mathcal{D} := \{(x_n, y_n) \in \mathcal{X} \times \mathcal{Y}\}_{n=1}^N$ where $y_n := y(x_n)$, and $y = y^\star + \epsilon$ where ϵ represents the noise as $\mathbb{E}(\epsilon|x) = 0$, et $\mathbb{V}(\epsilon) = \sigma^2$. We want to create an estimator $\hat{y}_\mathcal{D} : \mathcal{X} \to \mathcal{Y}$ which minimises the empirical risk in order to minimise the true risk $\mathcal{R}(\hat{y}) = \mathbb{E}_{(x,y)\sim\mathcal{P}}((y - \hat{y}(x))^2)$. Working in the context of [1]

Source Code

The reviewed source code and documentation for this algorithm are available from the web page of this article[1]. Usage instruction are included in the README.txt file of the archive.

1 Model

- $X = [x_1, \cdots, x_N]^T \in \mathcal{M}_{N,D}(\mathbb{R})$, $Y = [y_1, \cdots y_N]^T$
- $\hat{y}_\beta(x) = W_L \circ \sigma \circ \cdots \sigma \circ W_1(x)$, $W_l(x_l) = A_l x_l + b_l$
- $\sigma(x) = max(0, x)$ ReLU function.
- Constant-step gradient descent: $\hat{\beta}_{t+1} = \hat{\beta}_t - \alpha \nabla \hat{\mathcal{R}}(\hat{y}_{\hat{\beta}_t})$, $\alpha \in \mathbb{R}^+$
- Empirical Risk function $\hat{\mathcal{R}}(y) = \frac{1}{2}||Y - \hat{y}_\beta(X)||_2^2$

[1] https://ipolcore.ipol.im/demo/clientApp/demo.html?id=77777000527

2 Algorithms

Algorithm 1: Dataset initialisation

function dataset_initialisation*($f, C, M, ratio\_data$)*

- **Input** y, C, M, ratio_data**:** $y : \mathbb{R}^D \to \mathbb{R}$, $C = [[a_d, b_d], 1 \leq d \leq D]$, ratio_data $\in [0, 1]$
- $U = \mathcal{U}([0,1]^{(M,D)})$
- $X = Diag(a_d) + UDiag(b_d - a_d)$ # $X \hookrightarrow \mathcal{U}(C^M)$
- $Y = y(X)$
- $X_{train}, X_{test}, Y_{train}, Y_{test} = \text{test\_split}(X, Y, \text{ratio\_data})$
- **return** $\mathrm{X_{train}, X_{test}, Y_{train}, Y_{test}}$

Algorithm 2: MLP Gradient Descent

function MLP_Gradient_Descent*($\mathrm{P_{min}, P_{max}}$, Epochs, α, Data, Data_global)*

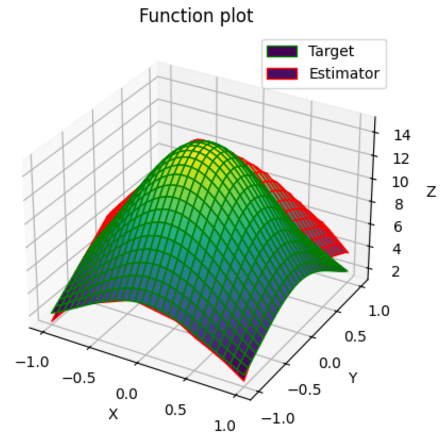
- $\mathrm{X_{train}, X_{test}, Y_{train}, Y_{test}} = \text{Data}$
- $\mathrm{X_{global}, Y_{global}} = \text{Data\_global}$
- **for** $P_{min} \leq p \leq P_{max}$ **do**
  - $MLP = Affine(1, P) \circ \sigma \circ Affine(P, D)$ # Creation of the MLP structure
  - $MLP.fit(X_{train}, Y_{train}, Epochs, \alpha)$
  - $\text{Train\_error[p]} = \log(1 + \mathrm{MSE}(\mathrm{Y_{train}}, \mathrm{MLP}(\mathrm{X_{train}})))$
  - $\text{Test\_error[p]} = \log(1 + \mathrm{MSE}(\mathrm{Y_{test}}, \mathrm{MLP}(\mathrm{X_{test}})))$
  - $\text{Global\_error[p]} = \log(1 + \mathrm{MSE}(\mathrm{Y_{test}}, \mathrm{MLP}(\mathrm{X_{global}})))$
- $\hat{y} = MLP(P_{max})$
- **return** Train_error, Test_error, Beta_norm, Global_error, $\hat{\mathrm{y}}$

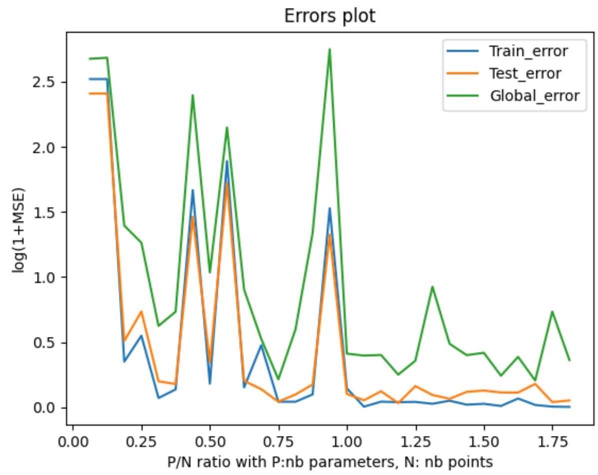
Algorithm 3: MLP Gradient Descent FIT

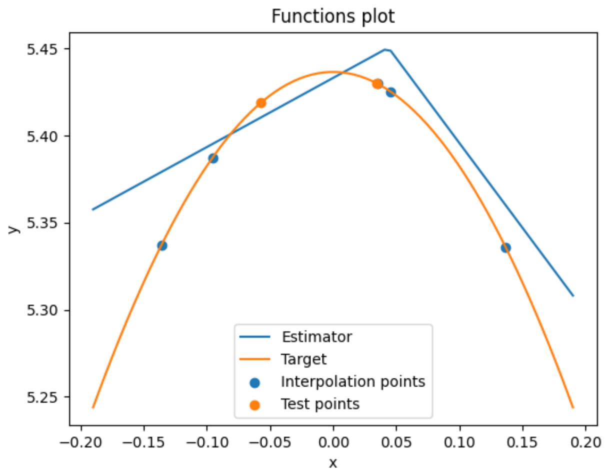
function MLP.fit*($\mathrm{X_{train}, Y_{train}}$, Epochs, α)*

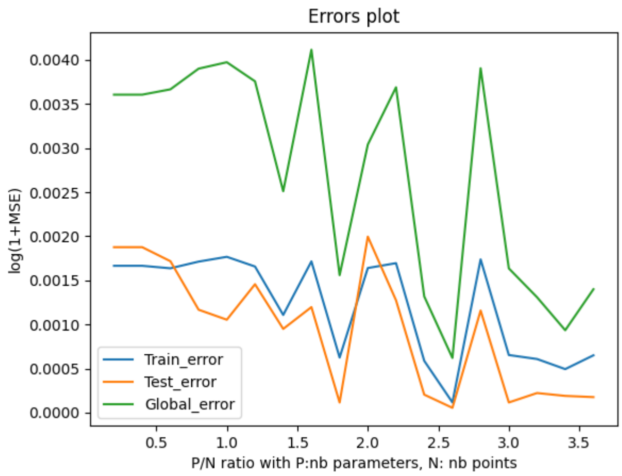
- **for** $1 \leq epoch \leq Epochs$ **do**
  - **for** $(x_{train}, y_{train}) \in \mathcal{D}$ **do**
    - $score\_gradient = MLP(x_{train}) - y_{train}$# gradient of MSE
    - $backpropagation(score\_gradient)$
  - $update(\alpha)$# udpate weights

3 Examples (random seed : 23334)

<table>
<tr><th>Function plot</th><th>Graph error</th></tr>
<tr>
<td>


$y(x) = 2e^{2-(x_1^2+x_2^2)},\ Epochs = 1000,\ P_{max} = 30$
</td>
<td>


$M = 20,\ r = 0.2,\ n = 30,\ a = 1,\ D = 2,\ \alpha = 0.001$
</td>
</tr>
<tr>
<td>


$y(x) = 3cos(40x) + cos(10x) - x^2,\ Epochs = 1997,\ P_{max} = 19$
</td>
<td>


$M = 7,\ r = 0.2,\ n = 80,\ a = 0.19,\ D = 1,\ \alpha = 0.1$
</td>
</tr>
</table>

References

[1] Emett Haddad, *Github Double Descent.* https://github.com/EmettGabrielH/Double-descent---Emett-Haddad, 2024. [Online].