PTML 7: 03/06/2022

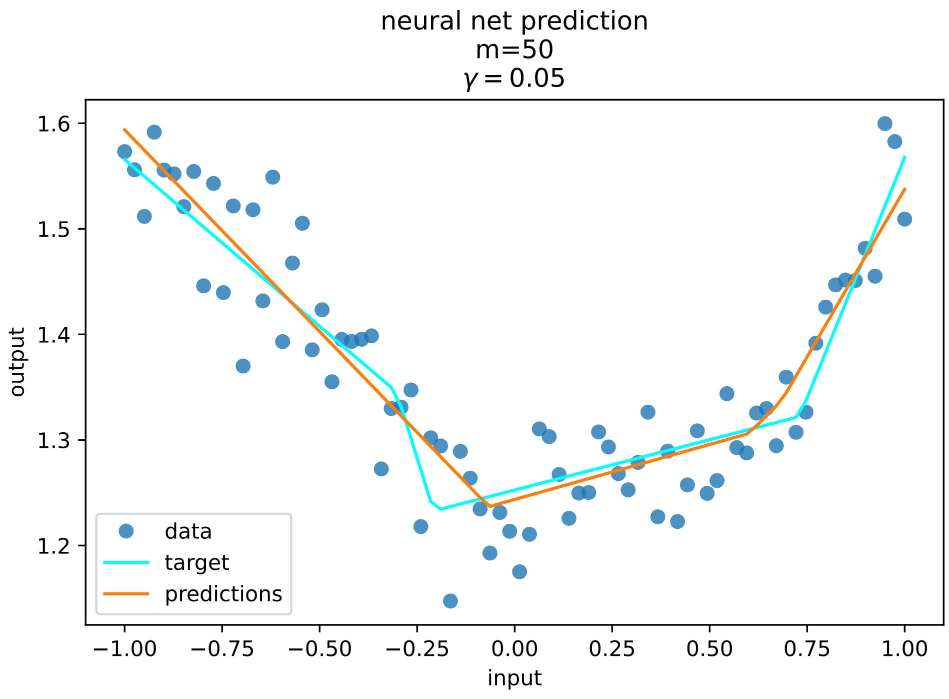


TABLE DES MATIÈRES

1 PTML 6

Many additional explanations and details, and a reference, have been added to the instructions of the previous TP. You can keep working on this one and experiment with the algorithm.

2 SIMPLICITY BIAS OF NEURAL NETWORKS

With some neural networks, it is unlikely to overfit the data. We will illustrate this with neurons that have ReLU activations. In this exercice, you can create new files taking blocks from the previous session.

To have some visual setting, we will set

— $\mathcal{X} = \mathbb{R}$

— $\mathcal{Y} = \mathbb{R}$

2.1 Target function generation

Exercice 1 : Generate a target function with a neural network that has $m = 5$ hidden layer with the same architecture has in TP6, but with a one dimensional input and ReLU activations. Generate a dataset by adding some noise to the outputs of this target function. See figure for an example function.

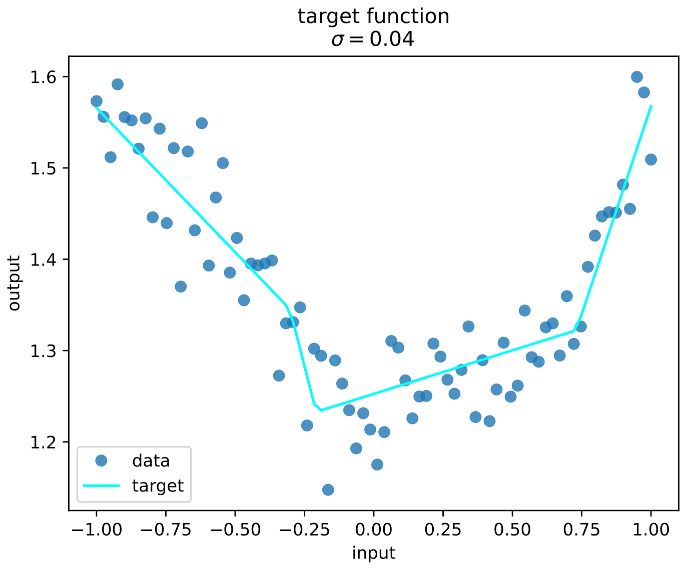


FIGURE 1 – Example target function and dataset

2.2 Network learning

Train a neural network with a varying number m of hidden layer, and with an initialization as follows :

— θ is initialized uniformly in $[-\frac{1}{\sqrt{m}}, -\frac{1}{\sqrt{m}}]^{m+1}$

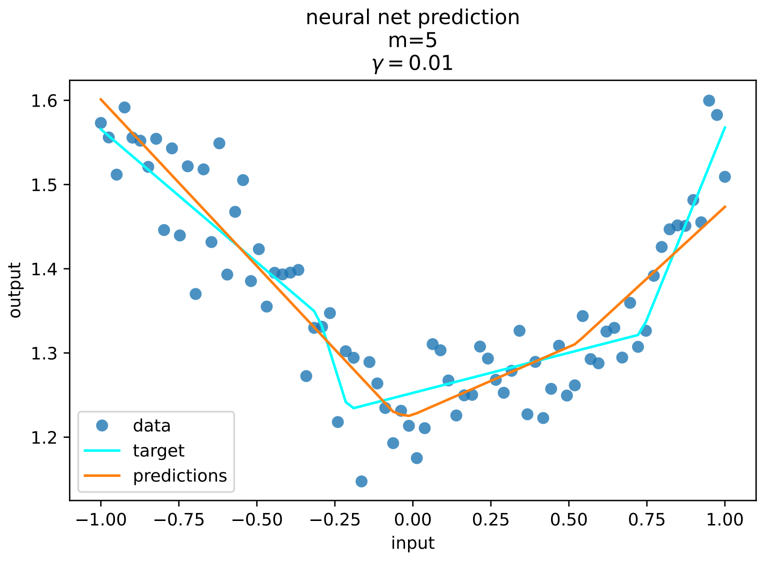
— Each column of w_h, that belongs to $\mathbb{R}^2$, is initialized on the sphere of radius $\frac{1}{\sqrt{m}}$.

You will probably need to look for small values of γ, in order to observe learning and the simplicity bias. Here are some example results.

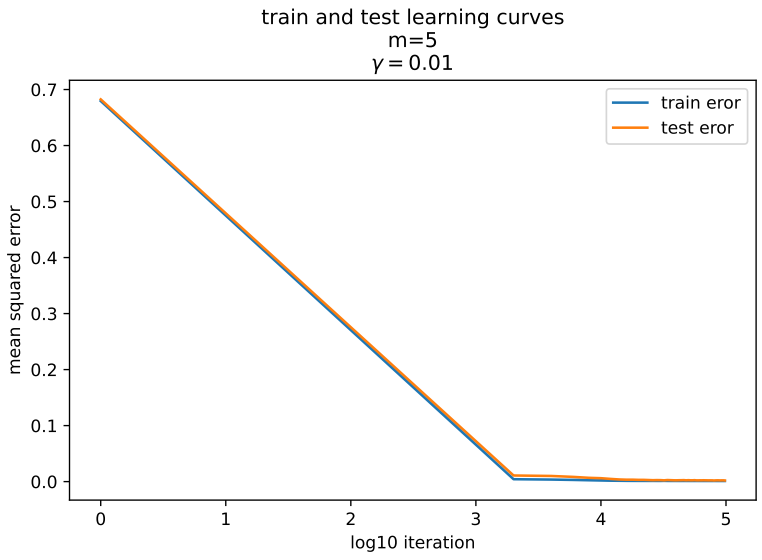
For the derivative of ReLU, you can use the heaviside function.

https://numpy.org/doc/stable/reference/generated/numpy.heaviside.html

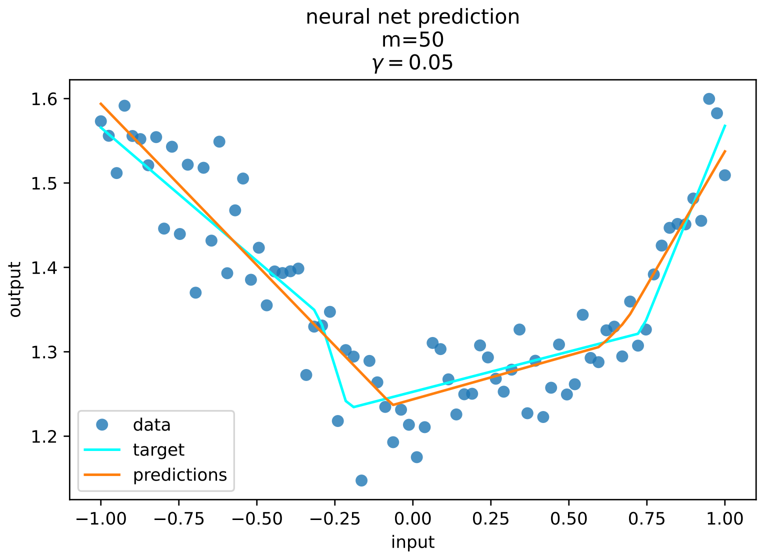
https://numpy.org/doc/stable/reference/generated/numpy.maximum.html



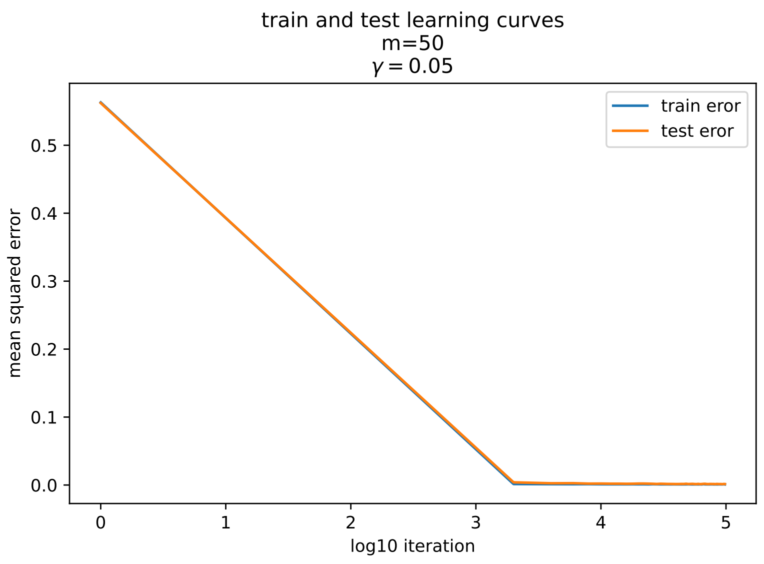
neural net prediction
m=5
$\gamma = 0.01$
output
input
data
target
predictions
1.6
1.5
1.4
1.3
1.2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00



train and test learning curves
m=5
$\gamma = 0.01$
mean squared error
log10 iteration
train eror
test eror
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
1
2
3
4
5



neural net prediction
m=50
$\gamma = 0.05$
output
input
data
target
predictions
1.6
1.5
1.4
1.3
1.2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00



2.3 Conclusion

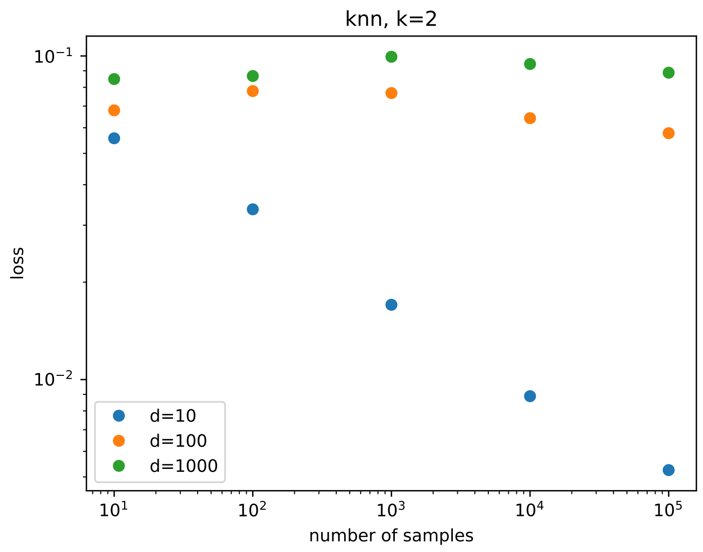
These neurons tend to not overfit, although some of them have a number of parameters way larger than of the minimal space containing the target function.

See more in this post.

https://francisbach.com/quest-for-adaptivity/

3 CURSE OF DIMENSIONALITY AND ADAPTIVITY TO A LOW DIMENSIONAL SUPPORT

Run a simulation that illustrates the curse of dimensionality by computing the error of a nearest neighbor estimator of f^*, for f^* being the euclidean norm on $\mathbb{R}^d$, for several numbers of samples n and several dimensions d. For instance, you can observe a picture like the one in figure **??**.


knn, k=2
loss
number of samples
d=10
d=100
d=1000