TÉCNICO LISBOA

Aprendizagem 2022/23

Homework III – Group 104

I. Pen-and-paper

1) Please see Appendix (1) for script used for the auxiliar calculations.

2)

$$\hat{z}(x, w) = 7.045 + 4.641x + 1.967x^2 - 1.301x^3$$

$$\hat{z}(x_1, w) = 7.045 + 4.641 \times 0.8 + 1.967 \times 0.640 - 1.301 \times 0.512 = 11.3509$$

$$\hat{z}(x_2, w) = 12.3525$$

$$\hat{z}(x_3, w) = 13.1992$$

$$\hat{z}(x_4, w) = 13.8287$$

$$\hat{z}(x_5, w) = 14.1785$$

	$\phi_0(x)$	$\phi_1(x)$	$\phi_2(x)$	$\phi_3(x)$	z
x_1	1	0.8	0.64	0.512	24
x_2	1	1	1	1	20
x_3	1	1.2	1.44	1.728	10
x_4	1	1.4	1.96	2.744	13
x_5	1	1.6	2.56	4.096	12

3)

$$RMSE(\hat{z}, z) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(z_i - \hat{z}_i)^2}$$

$$= \sqrt{\frac{1}{5}\left((24-11.3509)^2 + (20-12.3525)^2 + (10-13.1992)^2 + (13-13.8287)^2 + (12-14.1785)^2\right)}$$

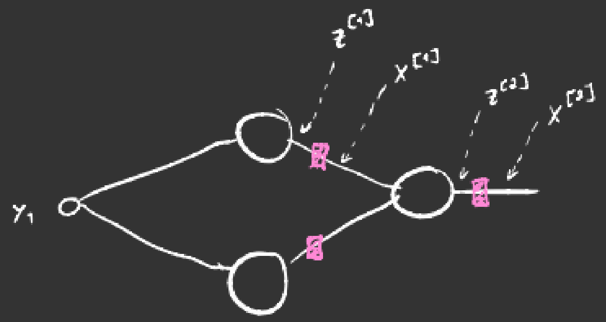
$$= \sqrt{46.8307}$$

$$= 6.84329$$

Aprendizagem 2022/23

Homework III – Group 104

4)



$f(x) = e^{0.1x}$

$\eta = 0.1$

$z^{[i]}$: pesos que saem da camada i antes de aplicação da função de ativação

$x^{[i]}$: pesos que saem da camada i após aplicação da função de ativação

Half squared error loss

$$E(w) = \frac{1}{2}\sum_{i=1}^{m}(z_i - \hat{z}_i)^2$$

$x_1 = 0.8$

→ FORWARD PROPAGATION

$$z^{[1]} = w^{[1]}x^{[0]} + b^{[1]}$$

$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix}\begin{bmatrix} 0.8 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.8 \\ 0.8 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1.8 \\ 1.8 \end{bmatrix}$$

$$x^{[1]} = f(z^{[1]}) = \begin{bmatrix} e^{0.1\times 1.8} \\ e^{0.1\times 1.8} \end{bmatrix} = \begin{bmatrix} 1.19722 \\ 1.19722 \end{bmatrix}$$

$$z^{[2]} = w^{[2]}x^{[1]} + b^{[2]}$$

$$= \begin{bmatrix} 1 & 1 \end{bmatrix}\begin{bmatrix} 1.19722 \\ 1.19722 \end{bmatrix} + \begin{bmatrix} 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2.39443 \end{bmatrix} + \begin{bmatrix} 1 \end{bmatrix}$$

$$= \begin{bmatrix} 3.39443 \end{bmatrix}$$

$$x^{[2]} = f(z^{[2]}) = \begin{bmatrix} e^{0.1\times 3.39443} \end{bmatrix} = \begin{bmatrix} 1.40417 \end{bmatrix}$$

$$\frac{\partial E}{\partial w^{[i]}} = \underbrace{\frac{\partial E}{\partial x^{[i]}} \circ \frac{\partial x^{[i]}}{\partial z^{[i]}}}_{\delta^{[2]}} \cdot \underbrace{\left(\frac{\partial z^{[i]}}{\partial w^{[i]}}\right)^T}_{(x^{[i-1]})^T}$$

$$x_1 = 0.8$$

$$z_1 = \begin{bmatrix} 1.8 \\ 1.8 \end{bmatrix} \quad z_2 = \begin{bmatrix} 3.39443 \end{bmatrix}$$

$$x_1 = \begin{bmatrix} 1.19722 \\ 1.19722 \end{bmatrix} \quad x_2 = \begin{bmatrix} 1.40417 \end{bmatrix}$$

$$f'(x) = 0.1 e^{0.1x}$$

$$\delta^{[2]} = (x^{[2]} - t) \circ f'(z^{[2]}) = \left(\begin{bmatrix} 1.40417 \end{bmatrix} - \begin{bmatrix} 24 \end{bmatrix}\right) \circ \begin{bmatrix} 0.1 e^{0.1 \times 3.39443} \end{bmatrix}$$

$$= \begin{bmatrix} -22.5958 \end{bmatrix} \circ \begin{bmatrix} 0.140417 \end{bmatrix} = \begin{bmatrix} -3.17282 \end{bmatrix}$$

$$(x^{[1]})^T = \begin{bmatrix} 1.19722 & 1.19722 \end{bmatrix}$$

→ BACK PROPAGATION

- $$\frac{\partial E}{\partial w^{[2]}} = \begin{bmatrix} -3.17282 \end{bmatrix} \cdot \begin{bmatrix} 1.19722 & 1.19722 \end{bmatrix} = \begin{bmatrix} -3.79857 & -3.79857 \end{bmatrix}$$

- $$\frac{\partial E}{\partial w^{[1]}} = \underbrace{\frac{\partial E}{\partial x^{[1]}} \circ \frac{\partial x^{[1]}}{\partial z^{[1]}}}_{\delta^{[1]}} \cdot \underbrace{\left(\frac{\partial z^{[1]}}{\partial w^{[1]}}\right)^T}_{(x^{[0]})^T} = \begin{bmatrix} -0.379856 \\ -0.379856 \end{bmatrix} \cdot \begin{bmatrix} 0.8 \end{bmatrix} = \begin{bmatrix} -0.303884 \\ -0.303884 \end{bmatrix}$$

$$\delta^{[1]} = \left(\underbrace{\left(\frac{\partial z^{[2]}}{\partial x^{[1]}}\right)^T}_{(w^{[2]})^T} \cdot \delta^{[2]}\right) \circ \underbrace{\frac{\partial x^{[1]}}{\partial z^{[1]}}}_{f'(z^{[1]})} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} -3.17282 \end{bmatrix} \circ \begin{bmatrix} 0.1 e^{0.1 \times 1.8} \\ 0.1 e^{0.1 \times 1.8} \end{bmatrix}$$

$$= \begin{bmatrix} -3.17282 \\ -3.17282 \end{bmatrix} \circ \begin{bmatrix} 0.1 e^{0.1 \times 1.8} \\ 0.1 e^{0.1 \times 1.8} \end{bmatrix} = \begin{bmatrix} -0.379856 \\ -0.379856 \end{bmatrix}$$

- $$\frac{\partial E}{\partial b^{[2]}} = \underbrace{\frac{\partial E}{\partial x^{[2]}} \circ \frac{\partial x^{[2]}}{\partial z^{[2]}}}_{\delta^{[2]}} \cdot \underbrace{\frac{\partial z^{[2]}}{\partial b^{[2]}}}_{1} = \begin{bmatrix} -3.17282 \end{bmatrix} \cdot \begin{bmatrix} 1 \end{bmatrix} = \begin{bmatrix} -3.17282 \end{bmatrix}$$

- $$\frac{\partial E}{\partial b^{[1]}} = \delta^{[1]} = \begin{bmatrix} -0.379856 \\ -0.379856 \end{bmatrix}$$

$$\delta^{[2]} = \begin{bmatrix} -3.17282 \end{bmatrix}$$

$$\delta^{[1]} = \begin{bmatrix} -0.379856 \\ -0.379856 \end{bmatrix}$$

$$\frac{\partial E}{\partial w^{[2]}} = \begin{bmatrix} -3.79857 & -3.79857 \end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[2]}} = \begin{bmatrix} -3.17282 \end{bmatrix}$$

$$\frac{\partial E}{\partial w^{[1]}} = \begin{bmatrix} -0.303884 \\ -0.303884 \end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[1]}} = \begin{bmatrix} -0.379856 \\ -0.379856 \end{bmatrix}$$

$x_2 = 1$

→ FORWARD PROPAGATION

$$z^{[1]} = \begin{bmatrix}1\\1\end{bmatrix}[1] + \begin{bmatrix}1\\1\end{bmatrix} = \begin{bmatrix}2\\2\end{bmatrix}$$

$$x^{[1]} = f(z^{[1]}) = \begin{bmatrix}e^{0.1\times2}\\e^{0.1\times2}\end{bmatrix} = \begin{bmatrix}1.2214\\1.2214\end{bmatrix}$$

$$z^{[2]} = \begin{bmatrix}1 & 1\end{bmatrix}\begin{bmatrix}1.2214\\1.2214\end{bmatrix} + [1] = [3.44281]$$

$$x^{[2]} = f(z^{[2]}) = \left[e^{0.1\times 3.44281}\right] = [1.41097]$$

→ BACK PROPAGATION

$$\frac{\partial E}{\partial w^{[2]}} = \underbrace{\frac{\partial E}{\partial x^{[2]}} \circ \frac{\partial x^{[2]}}{\partial z^{[2]}}}_{\delta^{[2]}} \cdot \underbrace{\left(\frac{\partial z^{[2]}}{\partial w^{[2]}}\right)^T}_{(x^{[1]})^T}$$

- $$\frac{\partial E}{\partial w^{[2]}} = (x^{[2]} - t) \circ f'(z^{[2]}) \cdot (x^{[1]})^T$$
$$= \left([1.41097] - [20]\right) \circ \left[0.1\,e^{0.1\times3.44281}\right] \cdot \begin{bmatrix}1.2214 & 1.2214\end{bmatrix}$$
$$= [-18.589] \circ [0.141098] \cdot \begin{bmatrix}1.2214 & 1.2214\end{bmatrix}$$
$$= [-2.62286] \cdot \begin{bmatrix}1.2214 & 1.2214\end{bmatrix}$$
$$= \begin{bmatrix}-3.20356 & -3.20356\end{bmatrix}$$

- $$\frac{\partial E}{\partial b^{[2]}} = \delta^{[2]} = [-2.62286]$$

- $$\frac{\partial E}{\partial w^{[1]}} = \delta^{[1]} \cdot (x^{[0]})^T = \left[\left((w^{[2]})^T \cdot \delta^{2}\right) \circ f'(z^{[1]})\right] \cdot (x^{[0]})^T$$
$$= \left[\left(\begin{bmatrix}1\\1\end{bmatrix} \cdot [-2.62286]\right) \circ \begin{bmatrix}0.1\,e^{0.1\times2}\\0.1\,e^{0.1\times2}\end{bmatrix}\right] \cdot [1]$$
$$= \left(\begin{bmatrix}-2.62286\\-2.62286\end{bmatrix} \circ \begin{bmatrix}0.12214\\0.12214\end{bmatrix}\right) \cdot [1]$$
$$= \begin{bmatrix}-0.320357\\-0.320357\end{bmatrix} \cdot [1] = \begin{bmatrix}-0.320357\\-0.320357\end{bmatrix}$$

- $$\frac{\partial E}{\partial b^{[1]}} = \delta^{[1]} = \begin{bmatrix}-0.320357\\-0.320357\end{bmatrix}$$

$$\delta^{[2]} = [-2.62286]$$
$$\delta^{[1]} = \begin{bmatrix}-0.320357\\-0.320357\end{bmatrix}$$

$$\frac{\partial E}{\partial w^{[2]}} = \begin{bmatrix}-3.20356 & -3.20356\end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[2]}} = [-2.62286]$$

$$\frac{\partial E}{\partial w^{[1]}} = \begin{bmatrix}-0.320357\\-0.320357\end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[1]}} = \begin{bmatrix}-0.320357\\-0.320357\end{bmatrix}$$

$x_3 = 1.2$ → FORWARD PROPAGATION

$$z^{[1]} = W^{[1]} x^{[0]} + b^{[1]} = \begin{bmatrix}1\\1\end{bmatrix}[1.2] + \begin{bmatrix}1\\1\end{bmatrix} = \begin{bmatrix}2.2\\2.2\end{bmatrix}$$

$$x^{[1]} = f(z^{[1]}) = \begin{bmatrix}e^{0.1\times 2.2}\\e^{0.1\times 2.2}\end{bmatrix} = \begin{bmatrix}1.24608\\1.24608\end{bmatrix}$$

$$z^{[2]} = \begin{bmatrix}1 & 1\end{bmatrix}\begin{bmatrix}1.24608\\1.24608\end{bmatrix} + [1] = [3.49215]$$

$$x^{[2]} = f(z^{[2]}) = [e^{0.1\times 3.49215}] = [1.41795]$$

→ BACK PROPAGATION

$$\frac{\partial E}{\partial W^{[2]}} = \underbrace{\frac{\partial E}{\partial x^{[2]}} \circ \frac{\partial x^{[2]}}{\partial z^{[2]}}}_{\delta^{[2]}} \cdot \underbrace{\left(\frac{\partial z^{[2]}}{\partial W^{[2]}}\right)^T}_{(x^{[1]})^T}$$

- $$\frac{\partial E}{\partial W^{[2]}} = (x^{[2]} - t) \circ f'(z^{[2]}) \cdot (x^{[1]})^T$$
$$= [1.41795] - [10] \circ [0.1\, e^{0.1\times 3.49215}] \cdot [1.24608 \quad 1.24608]$$
$$= [-8.58205] \circ [0.141795] \cdot [1.24608 \quad 1.24608]$$
$$= [-1.2169] \cdot [1.24608 \quad 1.24608]$$
$$= [-1.51635 \quad -1.51635]$$

- $$\frac{\partial E}{\partial b^{[2]}} = \delta^{[2]} = [-1.2169]$$

- $$\frac{\partial E}{\partial W^{[1]}} = \delta^{[1]} \cdot (x^{[0]})^T = \left[\left((W^{[2]})^T \cdot \delta^{2}\right) \circ f'(z^{[1]})\right] \cdot (x^{[0]})^T$$
$$= \left[\left(\begin{bmatrix}1\\1\end{bmatrix} \cdot [-1.2169]\right) \circ \begin{bmatrix}0.1\, e^{0.1\times 2.2}\\0.1\, e^{0.1\times 2.2}\end{bmatrix}\right] \cdot [1.2]$$
$$= \left(\begin{bmatrix}-1.2169\\-1.2169\end{bmatrix} \circ \begin{bmatrix}0.124608\\0.124608\end{bmatrix}\right) \cdot [1.2]$$
$$= \begin{bmatrix}-0.151635\\-0.151635\end{bmatrix} \cdot [1.2] = \begin{bmatrix}-0.181962\\-0.181962\end{bmatrix}$$

- $$\frac{\partial E}{\partial b^{[1]}} = \delta^{[1]} = \begin{bmatrix}-0.151635\\-0.151635\end{bmatrix}$$

$$\delta^{[2]} = [-1.2169]$$
$$\delta^{[1]} = \begin{bmatrix}-0.151635\\-0.151635\end{bmatrix}$$
$$\frac{\partial E}{\partial W^{[2]}} = [-1.51635 \quad -1.51635]$$
$$\frac{\partial E}{\partial b^{[2]}} = [-1.2169]$$
$$\frac{\partial E}{\partial W^{[1]}} = \begin{bmatrix}-0.181962\\-0.181962\end{bmatrix}$$
$$\frac{\partial E}{\partial b^{[1]}} = \begin{bmatrix}-0.151635\\-0.151635\end{bmatrix}$$

→ Atualização dos pesos:

$$w^{[1]} = w^{[1]} - \eta \sum \frac{\partial E}{\partial w^{[1]}} =$$

$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix} - 0.1 \left(\begin{bmatrix} -0.303884 \\ -0.303884 \end{bmatrix} + \begin{bmatrix} -0.320357 \\ -0.320357 \end{bmatrix} + \begin{bmatrix} -0.181962 \\ -0.181962 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 1.08062 \\ 1.08062 \end{bmatrix}$$

$$w^{[2]} = w^{[2]} - \eta \sum \frac{\partial E}{\partial w^{[2]}} =$$

$$= \begin{bmatrix} 1 & 1 \end{bmatrix} - 0.1 \left(\begin{bmatrix} -3.79857 & -3.79857 \end{bmatrix} + \begin{bmatrix} -3.20356 & -3.20356 \end{bmatrix} + \begin{bmatrix} -1.51635 & -1.51635 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 1.85185 & 1.85185 \end{bmatrix}$$

$$b^{[1]} = b^{[1]} - \eta \sum \frac{\partial E}{\partial b^{[1]}} =$$

$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix} - 0.1 \left(\begin{bmatrix} -0.379856 \\ -0.379856 \end{bmatrix} + \begin{bmatrix} -0.320357 \\ -0.320357 \end{bmatrix} + \begin{bmatrix} -0.151635 \\ -0.151635 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 1.08518 \\ 1.08518 \end{bmatrix}$$

$$b^{[2]} = b^{[2]} - \eta \frac{\partial E}{\partial b^{[2]}} =$$

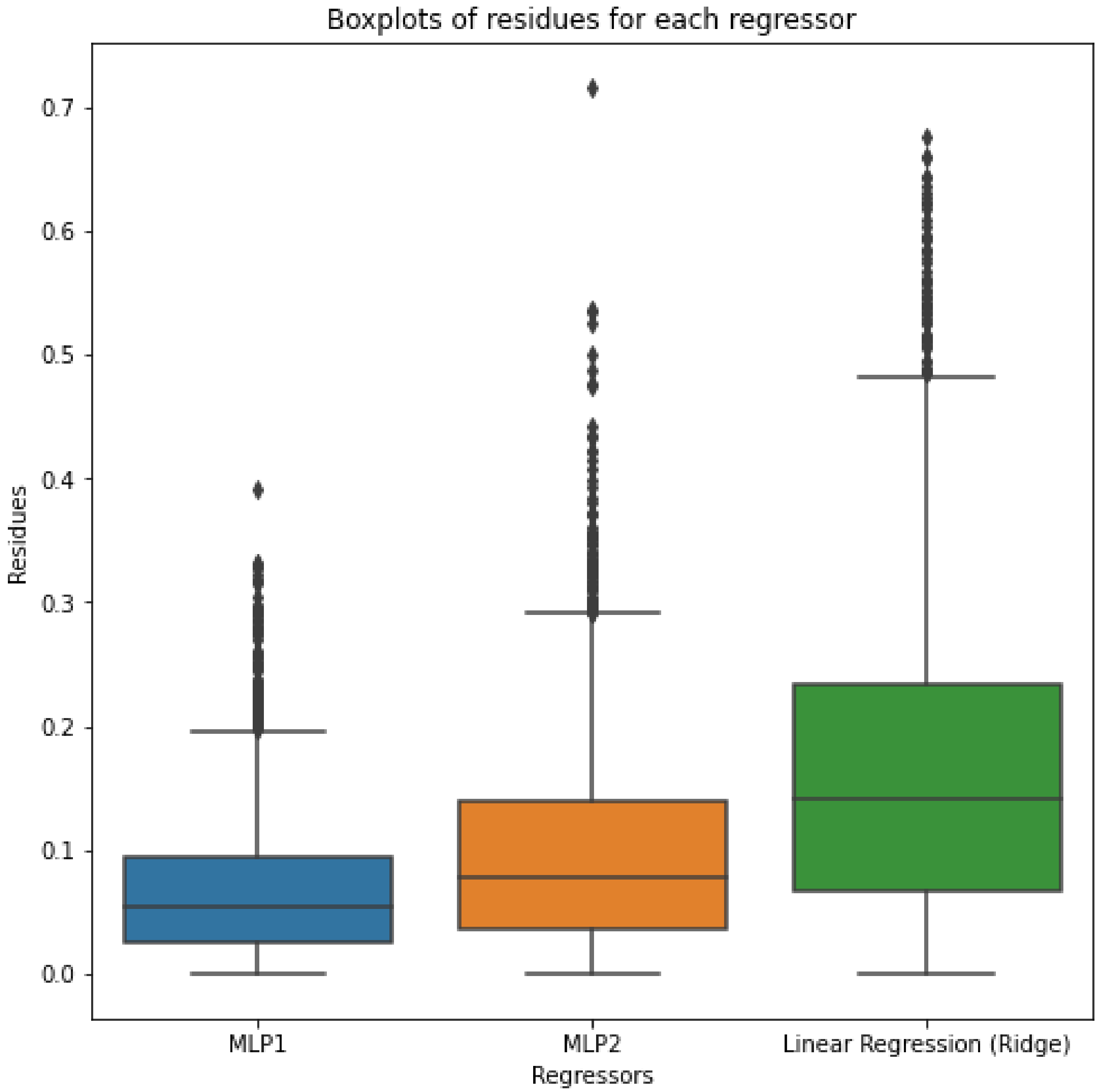
$$= \begin{bmatrix} 1 \end{bmatrix} - 0.1 \left(\begin{bmatrix} -3.17282 \end{bmatrix} + \begin{bmatrix} -2.62286 \end{bmatrix} + \begin{bmatrix} -1.2169 \end{bmatrix} \right)$$

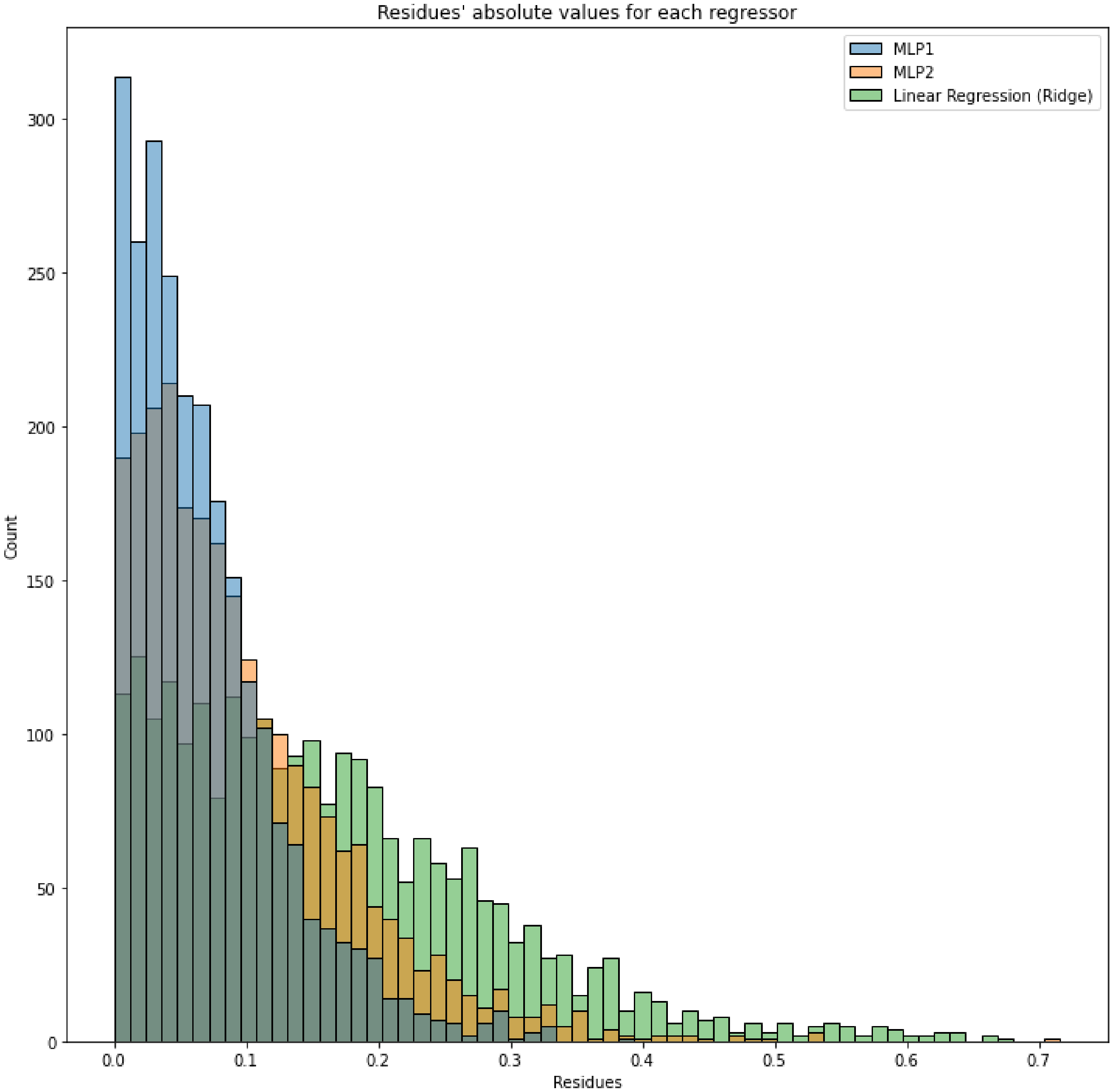
$$= \begin{bmatrix} 1.70126 \end{bmatrix}$$

II. Programming and critical analysis

5) Please see Appendix (2) for code.

6)





7)

```
print(f'MLP 1 converged after {mlp1.n_iter_} iterations.\nMLP 2 converged after
{mlp2.n_iter_} iterations.')
```

Output:

```
MLP 1 converged after 452 iterations.
MLP 2 converged after 77 iterations.
```

Aprendizagem 2022/23

Homework III – Group 104

8)

For the MLP1, there is a subdivision done in the original training set to obtain a validation set, that is used during the algorithm with early stopping. This algorithm not only keeps track of loss values but also keeps track of the validation scores at each iteration and stops when the validation score does not improve by a certain amount (*), for a certain number of iterations - consecutive epochs (**). With that said, if the validation score is constantly improving for a long time, that could explain the large number of iterations required for the MLP to converge, that is, the number of iterations after which the validation score starts to stabilize to a certain number (figure 2).

For the MLP2, no validation is done during training. Therefore, it looks exclusively to the loss value and, as soon as this value does not improve significantly, the process stops. That could explain why MLP2 converged way sooner than MLP1 (figure 1 and 2).

We set the argument `'verbose' to 'True'` in the MLRegression function, having obtained the loss values and the validation scores, at each epoch. We, then, used them to plot the variations of these metrics in figures 1 and 2 (presented in the next page). Bellow, we show the outputs for the last iteration of MLP1 and MLP2, respectively:

MLP1:

```
Iteration 452, loss = 0.00368074

Validation score: 0.890847

Validation score did not improve more than tol=0.000100 for 10 consecutive epochs.
Stopping.
```

MLP2:

```
Iteration 77, loss = 0.00768966

Training loss did not improve more than tol=0.000100 for 10 consecutive epochs.
Stopping.
```

(*) argument *'tol'* in *sklearn* (default=0.000100)

(**) argument *'n_iter_no_change'* in *sklearn* (default=10 epochs)

Aprendizagem 2022/23

Homework III – Group 104

Bellow, we present two line plots. The first one demonstrates the variation of loss values for each MLP. In the second one, we added the variation of validation scores underlying the early stopping of MLP1, to support our hypothesis regarding the differences in iterations between the two MLPs.

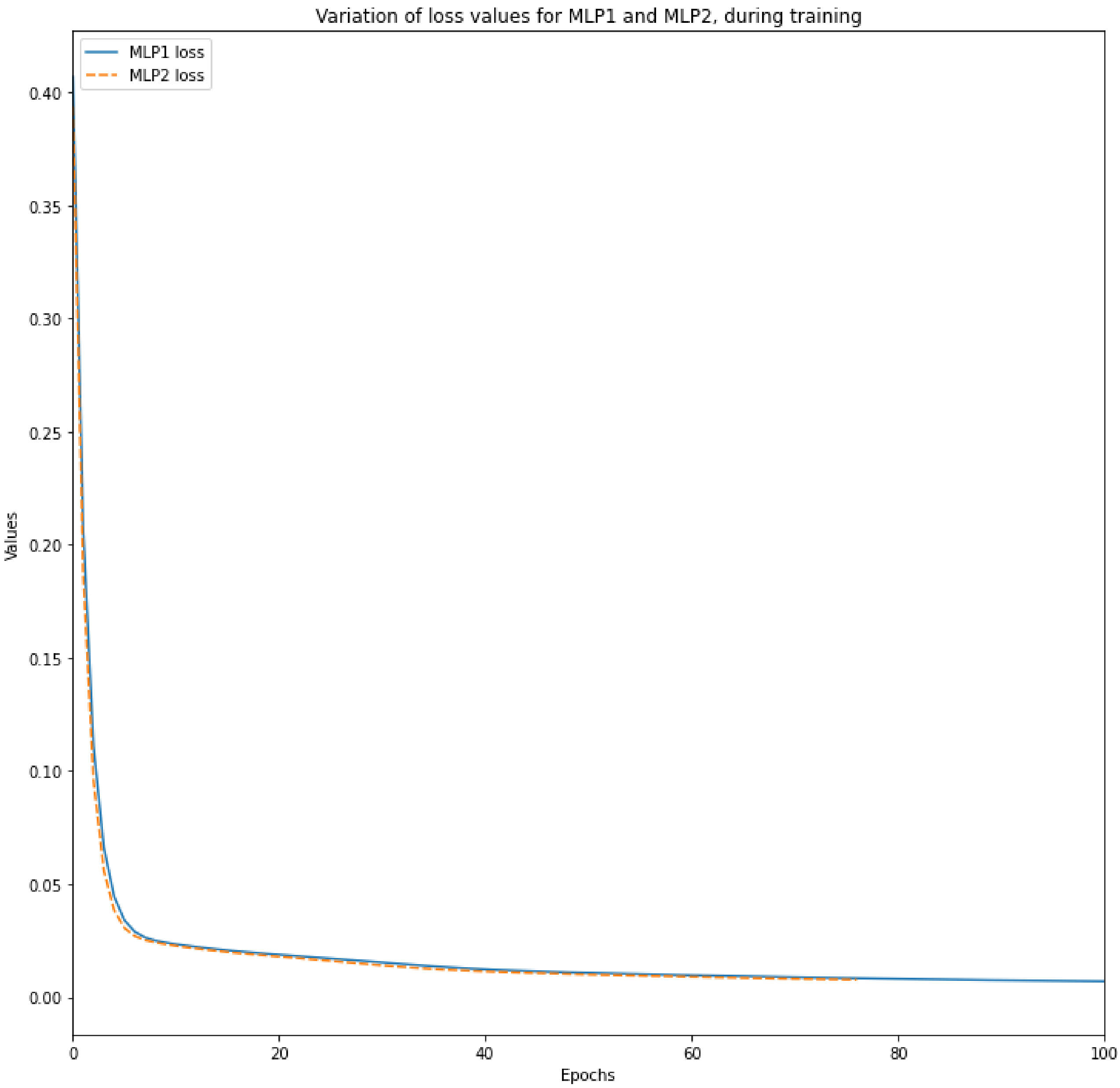


Figure 1

Above, we can see a considerable similarity between the loss variations between both MLPs, with those values stabilizing at about the same time. So, if these MLPs were looking exclusively to loss values, both would converge, sensibly, after a similar amount of number of iterations.

However, that is not the case, given the fact that MLP1's algorithm keeps track of the validation scores and relies on it to stop. That's when the following line comes in handy to help visualize what could be happening.

Note that, for simplicity and to help visualize the similarity between the lines, we've restrained the x-axis to 100, but bear in mind that the blue line continues until 452, but we show the full range in Figure 2.

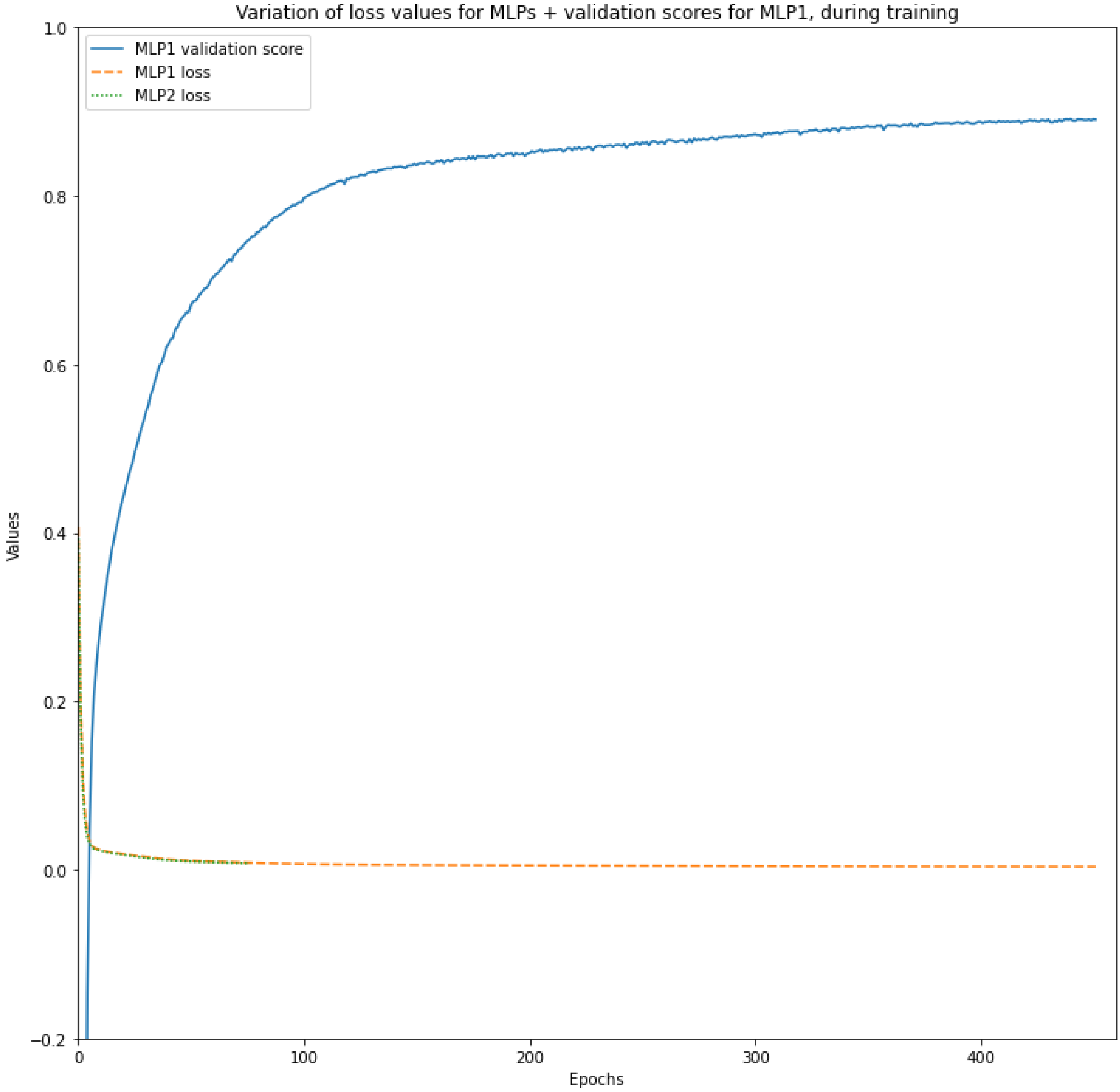


Figure 2

Here, we added the variation of validation scores regarding the early stopping of MLP1. As we can see, the line representing those values (blue one) takes a greater number of epochs to stabilize (converge) – around 452 iterations, compared to the loss variations, which tend to stabilize at a much smaller number of epochs – 77 iterations.

Note: to simplify, we omitted the negative values of the y axis regarding the MLP1 validation scores, since it does not serve the purpose of the question.

III. APPENDIX

(1)

```
import numpy as np

# pen and paper

def get_col_power(col, power):
    return [pow(x, power) for x in col]

def get_regression(x, y):

    yt = np.transpose(y)
    xt = np.transpose(x)

    x_xt = np.matmul(xt, x)
    sf_i = shrinkage_factor * np.identity(4)
    add1 = np.add(x_xt, sf_i)
    inv = np.linalg.pinv(add1)
    mul1 = np.matmul(inv, xt)
    w = np.matmul(mul1, yt)

    return w

shrinkage_factor = 2
col = [0.8, 1, 1.2, 1.4, 1.6]
arr = np.array([
    get_col_power(col, 0), get_col_power(col, 1), get_col_power(col, 2),
    get_col_power(col, 3)])
x = np.transpose(arr)
y = np.array([24, 20, 10, 13, 12])


reg = get_regression(x, y)
z_pred = []
n = 0
for el in col:
    z_el = 0
    n += 1
    for i in range(len(reg)):
        z_el += pow(el, i) * reg[i] if i != 0 else reg[i]
    z_pred.append(z_el)
    print(f'z_pred(x{n}, w) = {z_el}')

print(z_pred)
print(get_regression(x, y))
```

(2)

```
from scipy.io.arff import loadarff
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.neural_network import MLPRegressor
import pandas as pd, numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
from sklearn.model_selection import train_test_split


raw_data = loadarff('kin8nm.arff')
df = pd.DataFrame(raw_data[0])


X, y = df.drop("y", axis=1), df['y']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30,
random_state=0)
print("train:",X_train.shape,"\ntest:",X_test.shape)

ridge = Ridge(alpha=1.0).fit(X_train, y_train)
pred_test_ridge = ridge.predict(X_test)

mlp1 = MLPRegressor(hidden_layer_sizes=(10, 10), activation="tanh",
random_state=0, max_iter=500, early_stopping=True, verbose=True).fit(X_train,
y_train)
mlp2 = MLPRegressor(hidden_layer_sizes=(10, 10), activation="tanh",
random_state=0, max_iter=500, early_stopping=False, verbose=True).fit(X_train,
y_train)
pred_test_mlp1 = mlp1.predict(X_test)
pred_test_mlp2 = mlp2.predict(X_test)

# Get validations scores of MLP1 during training
validations_scores = []
fich = open('validation_scores.txt', 'r')
line = fich.readline()

while line:
    if 'Validation score: ' in line:
        validations_scores.append(float(line[18:-1]))
    line = fich.readline()

fich.close()
```

```python
# Get loss values of MLPs
mlp1_loss = mlp1.loss_curve_
mlp2_loss = mlp2.loss_curve_


# Lineplot with MLP's loss values
line_plot = sns.lineplot(data=[mlp1_loss, mlp2_loss])
plt.title('Variation of loss values for MLP1 and MLP2 during training')
plt.legend(loc='upper left', labels=['MLP1 loss', 'MLP2 loss'])
plt.xlabel('Epochs')
plt.ylabel('Values')
plt.xlim(0, 460)
plt.show()


# Lineplot with MLP's loss values + MLP1 validations scores
line_plot = sns.lineplot(data=[validations_scores, mlp1_loss, mlp2_loss])
plt.title('Variation of loss and validation scores during training')
plt.legend(loc='upper left', labels=['MLP1 validation score', 'MLP1 loss', 'MLP2 
loss'])
plt.xlabel('Epochs')
plt.ylabel('Values')
plt.xlim(0, 460)
plt.ylim(-0.2, 1)
plt.show()


y_true = y_test
print("Ridge MAE:",metrics.mean_absolute_error(y_true, pred_test_ridge))
print("MLP1 MAE:",metrics.mean_absolute_error(y_true, pred_test_mlp1))
print("MLP2 MAE:",metrics.mean_absolute_error(y_true, pred_test_mlp2))



residues_mlp1 = abs(y_test - pred_test_mlp1)
residues_mlp2 = abs(y_test - pred_test_mlp2)
residues_ridge = abs(y_test - pred_test_ridge)

df_residues = pd.DataFrame(list(zip(residues_mlp1, residues_mlp2, 
residues_ridge)), columns=['MLP1', 'MLP2', 'Linear Regression (Ridge)'])
```

```
# Boxplots of residues
plt.rcParams['figure.figsize'] = (8,8)
plt.title('Boxplots of residues for each regressor')
plt.ylabel('Residues')
plt.xlabel('Regressors')
sns.boxplot(data=df_residues)
plt.show()


# Histograms of residues
plt.rcParams['figure.figsize'] = (12, 12)
plt.title('Residues\' absolute values for each regressor')
plt.xlabel('Residues')
sns.histplot(data=df_residues)
plt.show()


raw_mse_mlp1 = metrics.mean_squared_error(y_true, pred_test_mlp1,
multioutput='raw_values')
raw_mse_mlp2 = metrics.mean_squared_error(y_true, pred_test_mlp2,
multioutput='raw_values')
raw_mse_ridge = metrics.mean_squared_error(y_true, pred_test_ridge,
multioutput='raw_values')


print(f'MLP 1 converged after {mlp1.n_iter_} iterations.\nMLP 2 converged after
{mlp2.n_iter_} iterations.')
```

END