Aprendizagem 2022/23

Homework III – Group 71

(ist1103471, ist1104139)

I. Pen-and-paper

1)a) Computing ϕ_1, ϕ_2 and ϕ_3:

$$\phi_1(x_1) = \exp\left(-\frac{\left\|\begin{pmatrix}0,7\\-0,3\end{pmatrix}-\begin{pmatrix}0\\0\end{pmatrix}\right\|^2}{2}\right) = \exp\left(-\frac{\sqrt{0,7^2+(-0,3)^2}^2}{2}\right) \approx 0,74826$$

$\phi_1(x_2) \approx 0,81165$

$\phi_1(x_3) \approx 0,71177$

$\phi_1(x_4) \approx 0,88250$

$\phi_2(x_1) \approx 0,74826$ $\quad$ $\phi_3(x_1) \approx 0,10127$

$\phi_2(x_2) \approx 0,27117$ $\quad$ $\phi_3(x_2) \approx 0,33121$

$\phi_2(x_3) \approx 0,09632$ $\quad$ $\phi_3(x_3) \approx 0,71177$

$\phi_2(x_4) \approx 0,16722$ $\quad$ $\phi_3(x_4) \approx 0,65377$

$$z = \begin{bmatrix}0,8\\0,6\\0,3\\0,3\end{bmatrix} \qquad \Phi = \begin{bmatrix}1 & \phi_1(x_1) & \phi_2(x_1) & \phi_3(x_1)\\1 & \phi_1(x_2) & \phi_2(x_2) & \phi_3(x_2)\\1 & \phi_1(x_3) & \phi_2(x_3) & \phi_3(x_3)\\1 & \phi_1(x_4) & \phi_2(x_4) & \phi_3(x_4)\end{bmatrix}$$

$$w = (\Phi^T \cdot \Phi + \lambda \cdot I)^{-1} \cdot \Phi^T \cdot z) = \begin{bmatrix}0,33914\\0,19945\\0,40096\\-0,29600\end{bmatrix}$$

Ridge Regression: $(x) = 0,33914 + 0,19945x_1 + 0,40096x_2 - 0,266x_3$

1b)

$$\hat{z} = \Phi w = \begin{bmatrix}0,75844\\0,51231\\0,30905\\-0,38263\end{bmatrix}$$

$$RMSE = \sqrt{\frac{1}{4}\sum_{i=1}^{4}(z-\hat{z})^2} \approx 0,06508$$

TÉCNICO LISBOA

Aprendizagem 2022/23

Homework III – Group 71

(ist1103471, ist1104139)

2) $\frac{\partial f(x)}{\partial x} = \frac{\partial}{\partial x}\tanh(0,5x-2) = 1-\tanh(0,5x-2)^2$

$x_A = \begin{bmatrix}1\\0\\0\\-1\end{bmatrix}$ $t_A = \begin{bmatrix}1\\-1\\-1\end{bmatrix}$ $x_B = \begin{bmatrix}1\\1\\1\\1\end{bmatrix}$ $t_B = \begin{bmatrix}-1\\1\\-1\end{bmatrix}$

$x^{[p]} = f(W^{[p]} x^{[p-1]} + b^{[p]})$

$X^{[p]} = f(z^{[p]})$

$z^{[1]A} = W^{[1]} x_A + b^{[1]} = \begin{bmatrix}1\\1\\1\end{bmatrix}$ $x^{[1]A} = \tanh(0,5 z^{[1]A} - 2) = \begin{bmatrix}-0,90515\\-0,90515\\-0,90515\end{bmatrix}$

$z^{[1]B} = W^{[1]} x_B + b^{[1]} = \begin{bmatrix}5\\6\\5\end{bmatrix}$ $x^{[1]B} = \begin{bmatrix}0,46212\\0,76159\\0,46212\end{bmatrix}$

$z^{[2]A} = W^{[2]} x^{[1]A} + b^{[2]} = \begin{bmatrix}-4,43090\\-1,71545\end{bmatrix}$ $x^{[2]A} = \begin{bmatrix}-0,99956\\-0,99343\end{bmatrix}$

$z^{[2]B} = W^{[2]} x^{[1]B} + b^{[2]} = \begin{bmatrix}4,97060\\2,68583\end{bmatrix}$ $x^{[2]B} = \begin{bmatrix}0,45048\\-0,57642\end{bmatrix}$

$z^{[3]A} = W^{[3]} x^{[2]A} + b^{[3]} = \begin{bmatrix}-0,99299\\-2,99211\\-0,99299\end{bmatrix}$ $x^{[3]A} = \begin{bmatrix}-0,98652\\-0,99816\\-0,98652\end{bmatrix}$

$z^{[3]B} = W^{[3]} x^{[2]B} + b^{[3]} = \begin{bmatrix}0,87406\\1,77502\\0,87406\end{bmatrix}$ $x^{[3]B} = \begin{bmatrix}-0,91590\\-0,80494\\-0,91590\end{bmatrix}$

- Elementary Rules:

$\frac{\partial E}{\partial x^{[p]}} = (x^{[p]} - t)$ $\frac{\partial x^{[p]}}{\partial z^{[p]}} = f'(z^{[p]}) = 1-\tanh(0,5 z^{[p]} - 2)^2$

$\frac{\partial z^{[p+1]}}{\partial x^{[p]}} = W^{[p+1]}$

- Deltas:

$\delta^{[3]A} = \frac{\partial E}{\partial x^{[3]A}} \circ \frac{\partial x^{[3]A}}{\partial z^{[3]A}} = (x^{[3]A} - t_A) \circ f'(z^{[3]A}) = \begin{bmatrix}-0,05320\\6,75280\times10^{-6}\\3,60994\times10^{-4}\end{bmatrix}$

$\delta^{[3]B} = (x^{[3]B} - t_B) \circ f'(z^{[3]B}) = \begin{bmatrix}0,01355\\-0,63546\\0,01355\end{bmatrix}$

$\delta^{[2]A} = \left(\frac{\partial z^{[3]A}}{\partial x^{[2]A}}\right)^T \cdot \delta^{[3]A} \circ f'(z^{[2]A}) = \begin{bmatrix}-4,59525\times10^{-5}\\-6,91571\times10^{-4}\end{bmatrix}$

$\delta^{[2]B} = \begin{bmatrix}-1,49792\\-0,40623\end{bmatrix}$

$\delta^{[1]A} = \begin{bmatrix}-1,33278\times10^{-4}\\-1,58190\times10^{-4}\\-1,33278\times10^{-4}\end{bmatrix}$ $\delta^{[1]B} = \begin{bmatrix}-1,49752\\-2,68693\\-1,49752\end{bmatrix}$

TÉCNICO LISBOA

Aprendizagem 2022/23

Homework III – Group 71

(ist1103471, ist1104139)

2) Updates!

$$\frac{\partial E}{\partial W^{[1]}} = \delta^{[1]A} (x_A)^T + \delta^{[1]B} (x_B)^T$$

$$W^{[1]new} = W^{[1]old} - \eta \frac{\partial E}{\partial W^{[1]}} = \begin{bmatrix} 1,14977 & 1,14975 & 1,14975 & 1,14974 \\ 1,26871 & 1,12687 & 2,26869 & 1,26868 \\ 1,14977 & 1,14975 & 1,14975 & 1,14974 \end{bmatrix} //$$

$$\frac{\partial E}{\partial b^{[1]}} = \delta^{[1]A} + \delta^{[1]B}$$

$$b^{[1]new} = b^{[1]old} - \eta \frac{\partial E}{\partial b^{[1]}} = \begin{bmatrix} 1,14977 \\ 1,26871 \\ 1,14977 \end{bmatrix} //$$

$$\frac{\partial E}{\partial W^{[2]}} = \delta^{[2]A} (x^{[1]A})^T + \delta^{[2]B} (x^{[1]B})^T$$

$$W^{[2]new} = W^{[2]old} - \eta \frac{\partial E}{\partial W^{[2]}} = \begin{bmatrix} 1,06922 & 4,11408 & 1,06922 \\ 1,01871 & 1,03088 & 1,01871 \end{bmatrix} //$$

$$\frac{\partial E}{\partial b^{[2]}} = \delta^{[2]A} + \delta^{[2]B}$$

$$b^{[2]new} = b^{[2]old} - \eta \frac{\partial E}{\partial b^{[2]}} = \begin{bmatrix} 1,14980 \\ 1,04069 \end{bmatrix} //$$

$$\frac{\partial E}{\partial W^{[3]}} = \delta^{[3]A} (x^{[2]A})^T + \delta^{[3]B} (x^{[2]B})^T$$

$$W^{[3]new} = W^{[3]old} - \eta \frac{\partial E}{\partial W^{[3]}} = \begin{bmatrix} 0,99407 & 0,99550 \\ 3,02863 & 0,96337 \\ 0,99943 & 1,00082 \end{bmatrix} //$$

$$\frac{\partial E}{\partial b^{[3]}} = \delta^{[3]A} + \delta^{[3]B}$$

$$b^{[3]new} = b^{[3]old} - \eta \frac{\partial E}{\partial b^{[3]}} = \begin{bmatrix} 1,00397 \\ 1,06355 \\ 0,99861 \end{bmatrix} //$$

Aprendizagem 2022/23

Homework III – Group 71

(ist1103471, ist1104139)

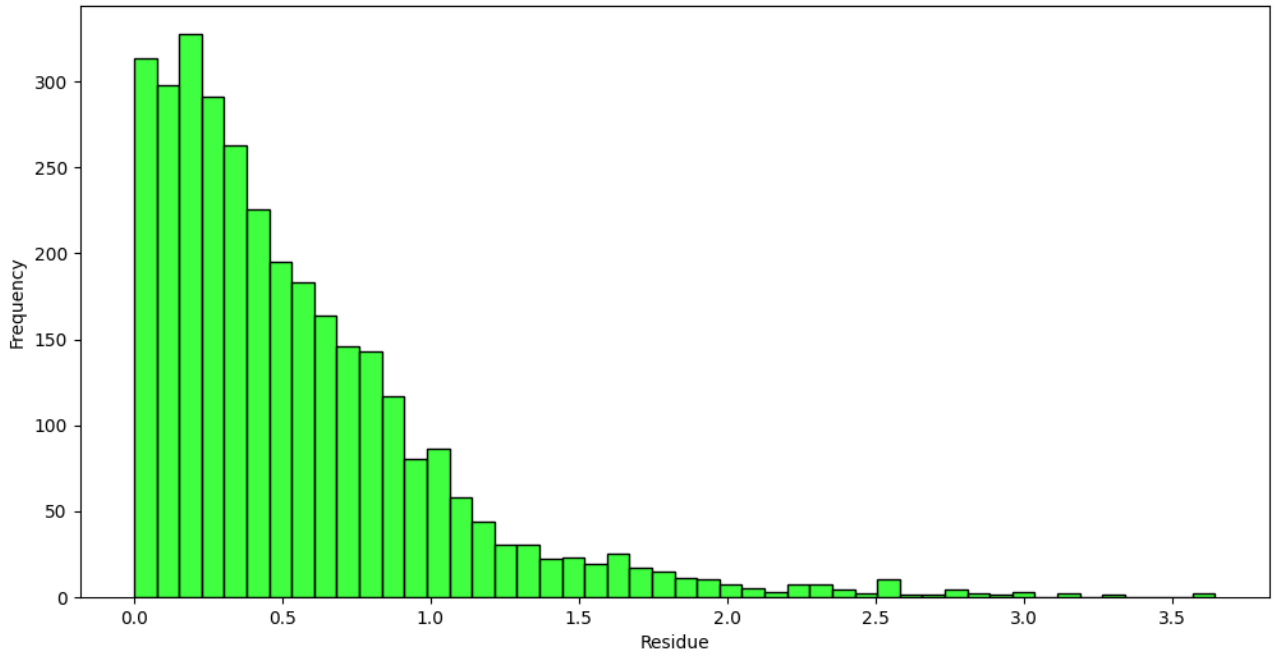
II. Programming and critical analysis

1)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPRegressor
from sklearn import metrics
from sklearn.exceptions import ConvergenceWarning

warnings.simplefilter("ignore", category=ConvergenceWarning)
data = pd.read_csv("winequality-red.csv", sep=";")
X = data.drop("quality", axis=1)
y = data["quality"]
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.8, random_state=0, stratify=y)

predictions = []
residue_list = []
MAE = []
RMSE = []
for seed in range(1, 11):
    mlp = MLPRegressor(hidden_layer_sizes=(10, 10), activation='relu',
                       random_state=seed, early_stopping=True, validation_fraction=0.2)
    mlp.fit(X_train, y_train)
    pred = mlp.predict(X_test)
    predictions.append(pred)
    residue = pred - y_test
    residue_list.append(np.abs(residue))
    MAE.append(metrics.mean_absolute_error(y_test, pred))
    RMSE.append(metrics.mean_squared_error(y_test, pred, squared=False))

residues = np.array(residue_list).flatten()
plt.figure(figsize=(12, 6))
sns.histplot(data=residues, color="lime")
plt.xlabel('Residue')
plt.ylabel('Frequency')
plt.show()
```

Homework III – Group 71

(ist1103471, ist1104139)



2)

```
def bound(x):
    rounded_x = round(x)
    if rounded_x < 0:
        return 0
    elif rounded_x > 10:
        return 10
    return rounded_x


bounded_MAE = []
for pred in predictions:
    new_pred = []
    for value in pred:
        new_pred.append(bound(value))
    bounded_MAE.append(metrics.mean_absolute_error(y_test, new_pred))

print("Mean of Bounded MAE's:", np.mean(bounded_MAE), "\nMean of Normal MAE's", np.mean(MAE))
```

```
Mean of Bounded MAE's: 0.4928125
Mean of Normal MAE's 0.5437511706983347
```

Since the mean absolute error is smaller than the one previously calculated with the unbounded estimates, we can conclude, that this technique does, indeed, help to get better results.

3)

```
mini_batch = [20, 50, 100, 200]
RMSE_20 = []
RMSE_50 = []
RMSE_100 = []
RMSE_200 = []


for seed in range(1, 11):
    for n in mini_batch:
        mlp = MLPRegressor(hidden_layer_sizes=(10, 10), activation='relu', random_state=seed,
                           max_iter=n, validation_fraction=0.2)
        mlp.fit(X_train, y_train)
        pred = mlp.predict(X_test)
        if n == 20:
            RMSE_20.append(metrics.mean_squared_error(y_test, pred, squared=False))
        elif n == 50:
            RMSE_50.append(metrics.mean_squared_error(y_test, pred, squared=False))
        elif n == 100:
            RMSE_100.append(metrics.mean_squared_error(y_test, pred, squared=False))
        else:
            RMSE_200.append(metrics.mean_squared_error(y_test, pred, squared=False))

print("Early Stopping RMSE:", np.mean(RMSE))
print("RMSE 20:            ", np.mean(RMSE_20))
print("RMSE 50:            ", np.mean(RMSE_50))
print("RMSE 100:           ", np.mean(RMSE_100))
print("RMSE 200:           ", np.mean(RMSE_200))
```

```
Early Stopping RMSE: 0.7285645002031444
RMSE 20:             1.5741537078556829
RMSE 50:             0.9296215581878042
RMSE 100:            0.7495674512113977
RMSE 200:            0.697002682560151
```

As we can see from the results, for a number of maximum iterations lower than 200, the root mean squared error is higher than the one calculated previously with an early stopping set to true. This means that the impact of such values for a max number of iterations is rather negative, since it means that the model is underfitted. On the other hand for a value of 200 the impact is good since the RMSE is lower.

4) By the results given in the last exercise, we can say that an early stop of the model's training, is overall good for the performance. None the less, we got better results by setting a well defined number of maximum iterations (200), this could happen because, the validation fraction does not represent the testing sample very well, meaning that the model's training will stop earlier than it would, if the validation sample was more similar to the testing sample.

END