KMGRYTTE - Katrine Marie Jaskiewicz Grytten

I reccommend running my program in Anaconda prompt. It can be called from the neccessary folder with

python movements.py

I've fixed the previous code. Now the code works as intended, although with the slight glitch that my sigmoid doesn't seem to be taking effect and the outputs exceed the range of -1 - 1. I've tried to fix it, but I wasn't able to maneuver around the digit limit of floats and doubles. The round(k,200) function I tried didn't seem to have an impact on the errors I got.

Nevertheless, it works as intended and I can observe the behaviour of a perceptron with it.

Here are my runs for hidden nodes 6, 12 and 20:

6:

finished in time:

9.293245553970337

[13, 0, 0, 0, 0, 0, 0, 0]

[0, 14, 0, 0, 0, 1, 0, 0] //Here b was confused with f

[0, 0, 11, 0, 0, 0, 0, 0]

[0, 2, 0, 14, 1, 0, 0, 1] //Here d was confused once for e and two times for b

[0, 0, 0, 0, 11, 0, 0, 0]

[0, 0, 0, 0, 0, 9, 0, 0]

[0, 1, 0, 0, 0, 3, 18, 0] //G was confused with f thrice and b once

[2, 0, 0, 0, 0, 0, 0, 10] //H was mistaken twice for a

classwise accuracy:

100.0

93.33333333333333

100.0

77.77777777777779

100.0

100.0

81.81818181818183

83.33333333333334

total accuracy:

92.03282828282828

12:

3.862457752227783

[10, 0, 0, 0, 1, 0, 0, 0] //A was mistaken once for e

[0, 19, 0, 0, 0, 0, 0, 0]

[0, 0, 11, 0, 0, 0, 0, 0]

[0, 0, 0, 13, 0, 0, 0, 0]

[1, 0, 0, 0, 13, 0, 0, 0] //E was mistaken once for a

[0, 1, 0, 0, 0, 12, 2, 0] //F was mistaken once for b and once for g

[0, 0, 0, 0, 0, 1, 12, 0] //G was mistaken once for f

[0, 0, 0, 0, 0, 0, 0, 15]

classwise accuracy:

90.9090909090909

100.0

100.0

100.0

92.85714285714286

80.0

92.3076923076923

100.0

total accuracy:

94.50924075924077

20:

finished in time:

17.111461400985718

[9, 0, 0, 0, 1, 0, 0, 0] //A was confused once for e

[0, 14, 0, 0, 0, 0, 0, 0]

[0, 0, 18, 0, 0, 0, 0, 0]

[0, 0, 0, 16, 1, 0, 1, 0] //D was confused once for e and g

[1, 0, 0, 1, 12, 0, 0, 0] //E was confused once for a and d

[0, 0, 0, 0, 0, 9, 0, 0]

[0, 0, 0, 0, 0, 1, 13, 0] //G was confused once for f

[0, 0, 0, 0, 0, 0, 0, 14]

classwise accuracy:

90.0

100.0

100.0

88.88888888888889

85.71428571428571

100.0

92.85714285714286

100.0

total accuracy:

94.68253968253968

20 nodes has varied very much in how much time is taken. Sometimes it has been faster than 12, but mostly it takes longer. 12 nodes seem to be the sweet spot since it has many dimensions so the weights

don't have to be adjusted to veery high precision, and it doesn't take ages to backpropagate through all of them. 6 couldn't even reach that level as it wasn't stopped by earlystoppings fitness check.

I've observed no obvious patterns as to which classes get confused for which consistently in multiple runs of the code, with the exception of one pattern in class d mostly being confused with e. Only one of my runs has given 100% accuracy for d. I haven't of course, run it enough times to prove a statistical significance. In these test runs a and e seem to slightly overlap