IA353A - Neural Networks
EFC1 - Question 1

Rafael Claro Ito
(R.A.: 118430)

April 2020

1 Source files

The Jupyter notebook with the code used to generate the plots and results presented in this report, all figures showed here and even the LaTeX source code used to generate this PDF can be found at the following GitHub repository:

`https://github.com/ito-rafael/IA353A-NeuralNetworks-1s2020`

2 Regularization coefficient (Ridge Regression)

2.1 Results summary

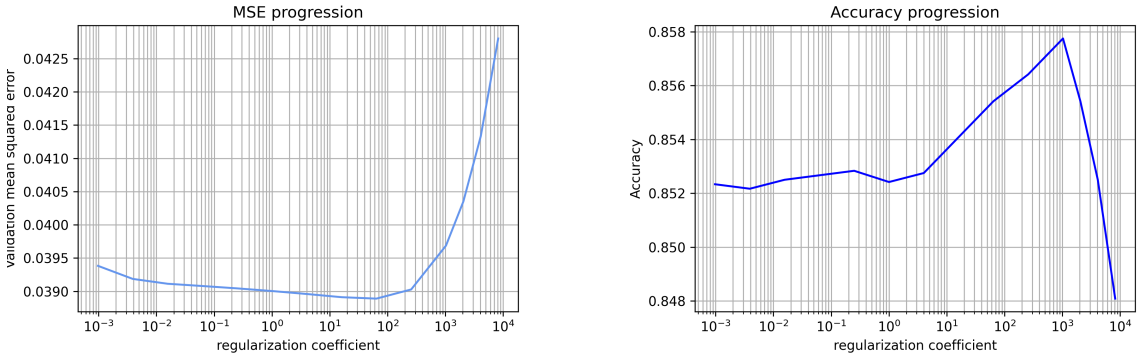
	λ optimum	
	MSE	Accuracy
coarse search	64	1024
fine search	51.5	1091.8

Table 1: Values of regularization coefficient found in coarse and fine searches

2.2 Coarse search

While performing the coarse search for the best regularization coefficient, 3 more values of lambda were added. This was done in order to see the falling of the accuracy curve, since the last suggested value, 2^{10}, had the best accuracy. The final values of lambda tested were:

$$\text{alpha\_interval} = [2^{-10}, 2^{-8}, 2^{-6}, 2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}, 2^{6}, 2^{8}, 2^{10}, \mathbf{2^{11}, 2^{12}, 2^{13}}]$$

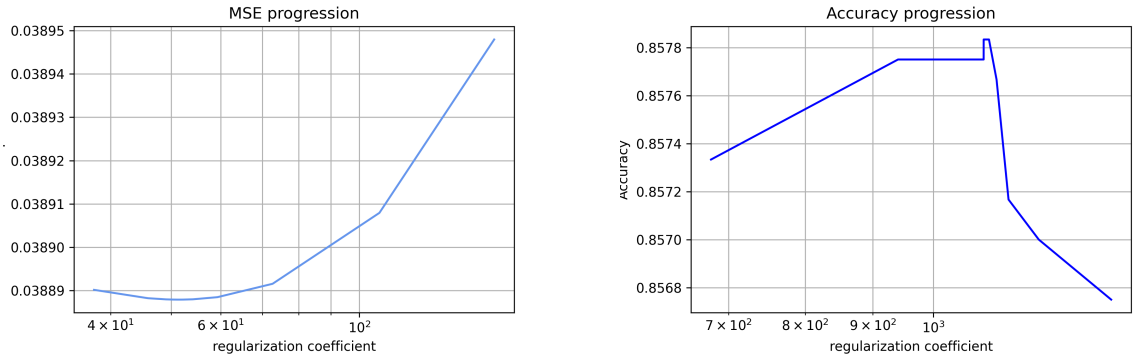


(a) MSE progression

(b) Accuracy progression

Figure 2: Progression of MSE and accuracy in validation set for different values of the regularization coefficient (coarse search)

2.3 Fine search



(a) MSE progression)

(b) Accuracy progression)

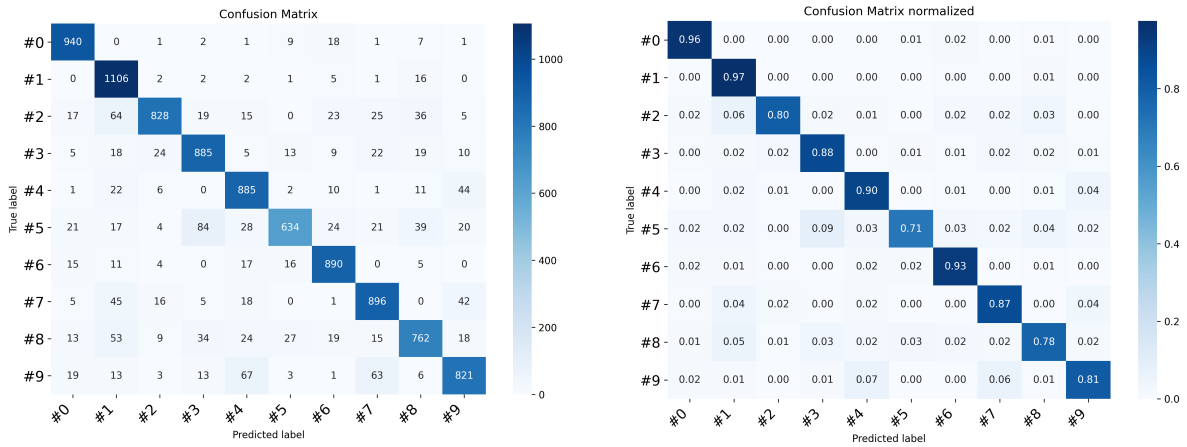
Figure 3: Progression of MSE and accuracy in validation set for different values of the regularization coefficient (fine search)

In order to perform the fine search of the regularization coefficient, a golden-section one dimensional search algorithm was coded. Among the function parameters, the most important ones are the intervals of the search, precision desired and the loss function. The code can be found in:
`https://github.com/ito-rafael/machine-learning/blob/master/snippets/golden_section_search_valid.py`

3 Confusion Matrix

Digit	nº of samples
0	980
1	1135
2	1032
3	1010
4	982
5	892
6	958
7	1028
8	974
9	1009

Table 2: Number of samples for each class in the test set



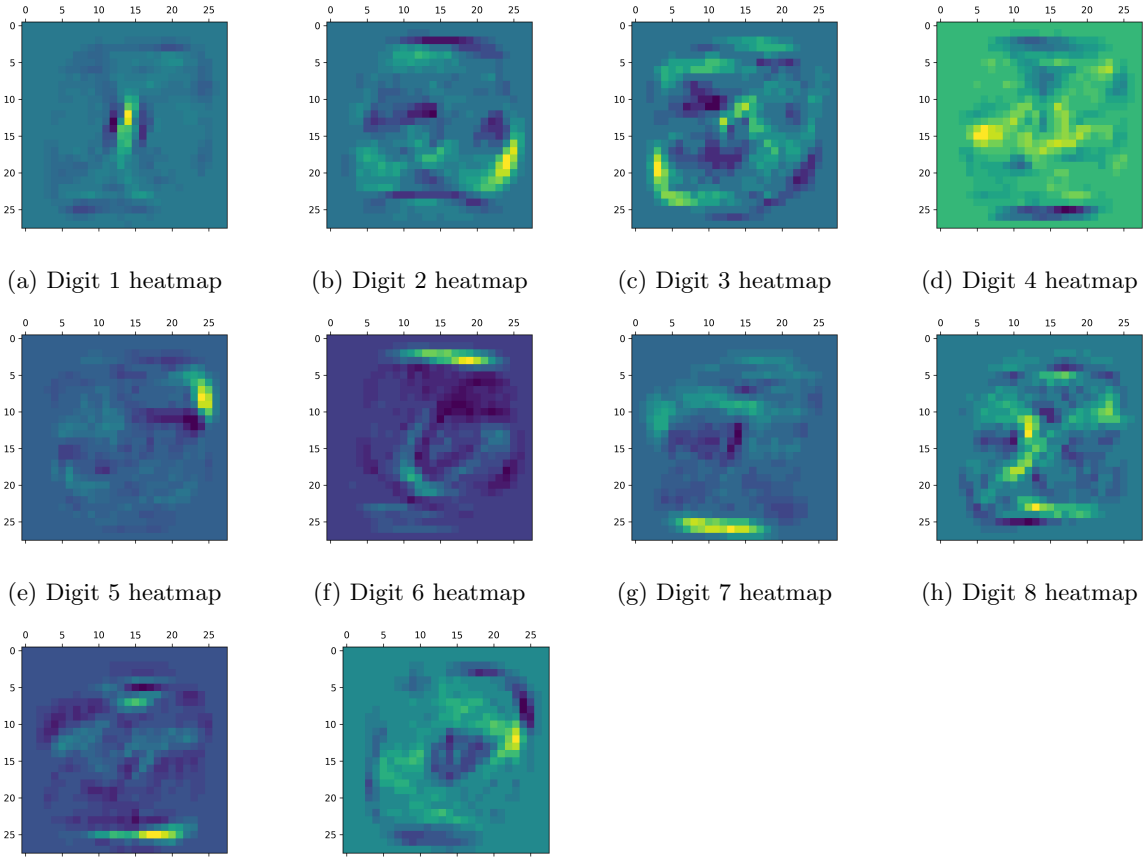
(a) Confusion matrix with raw values

(b) Confusion matrix with normalized values

Figure 4: Confusion matrix with normalized and raw values

The values displayed in the confusion matrix were obtained with the linear classifier applied in the test set. The test set is somewhat balanced, containing the number of samples for each class as illustrated in Figure 2

4 Classifier Heatmap



(a) Digit 1 heatmap

(b) Digit 2 heatmap

(c) Digit 3 heatmap

(d) Digit 4 heatmap

(e) Digit 5 heatmap

(f) Digit 6 heatmap

(g) Digit 7 heatmap

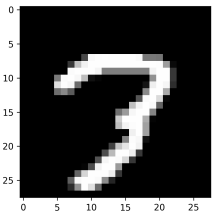
(h) Digit 8 heatmap

(i) Digit 9 heatmap

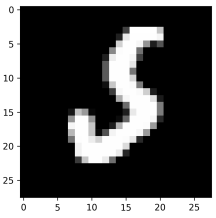
(j) Digit 0 heatmap

Figure 5: Heatmap for the classifiers correspondent to each digit

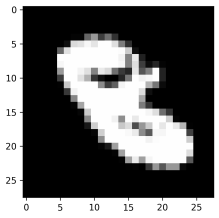
5 Misclassified data



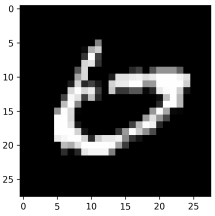
(a) Real class: 3 (0.12)
Predicted class: 7 (0.86)



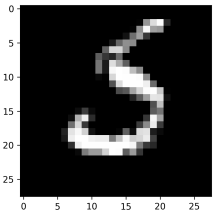
(b) Real class: 5 (-0.16)
Predicted class: 6 (0.57)



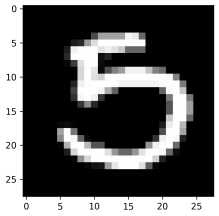
(c) Real class: 8 (-0.00)
Predicted class: 2 (0.72)



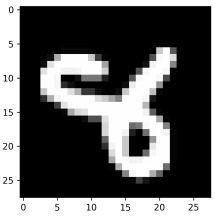
(d) Real class: 6 (0.06)
Predicted class: 0 (0.76)



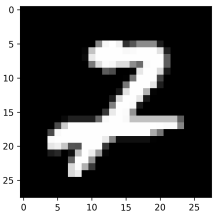
(e) Real class: 5 (-0.10)
Predicted class: 6 (0.58)



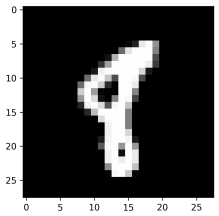
(f) Real class: 5 (0.04)
Predicted class: 0 (0.71)



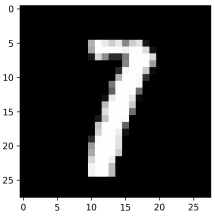
(g) Real class: 8 (0.16)
Predicted class: 4 (0.79)



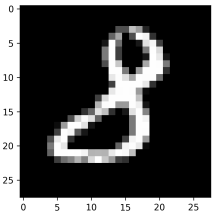
(h) Real class: 7 (-0.01)
Predicted class: 2 (0.63)



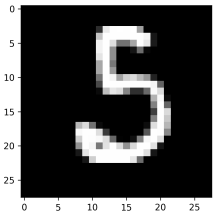
(i) Real class: 8 (0.12)
Predicted class: 1 (0.75)



(j) Real class: 1 (0.04)
Predicted class: 7 (0.64)



(k) Real class: 8 (-0.15)
Predicted class: 3 (0.53)



(l) Real class: 5 (-0.05)
Predicted class: 6 (0.58)

Figure 6: Misclassified digits

The digits shown in Figure 6 are in the top 30 misclassified digits from which the difference between the output for the real class and the output for the predicted class are the highest. Both real and predicted class and its outputs values associated are indicated. The output is shown in parenthesis, in front of the correspondent class.

6 Theoretical question

6.1 Question:

The choice of a different regularization coefficient for each class (instead of a single value for all classes) can give a higher performance?

6.2 Answer:

Obs.: The answer for this question will be written in Portuguese, due to the facility in expressing ideas related to this subject.

A forma de regularização que estamos tratando nessa questão refere-se a penalizações proporcionais à norma do vetor de parâmetros W. Sua principal função é a de evitar overfitting, aumentando a capacidade de generalização do modelo, e por consequência, a performance. Assim, o valor de lambda controla portanto o trade-off entre a redução do erro de aproximação e a limitação da magnitude dos parâmetros.

Tendo dito isso, passamos para a análise do problema de minimização envolvido. Na primeira forma (um único lambda para todas classes), queremos minimizar o somatório (para cada classe) do primeiro termo referente ao critério MSE somado com o segundo termo que é dado por lambda multiplicado pela norma do vetor de pesos. Na segunda forma (um lambda diferente para cada classe), temos no somatório o mesmo primeiro termo de antes, referente ao critério MSE, mas com o segundo termo tendo a norma do vetor de pesos multiplicada por lambdas diferentes.

Desta forma, o que difere a primeira proposta da segunda, é que a primeira tem o somatório de C=1 a C=10 para os valores de um mesmo lambda multiplicado pela coluna do vetor de pesos (de dimensão 785 x 1) associada a classe em questão, enquanto que na segunda proposta temos lambdas diferentes multiplicando as mesmas colunas do vetor de pesos W.

Podemos claramente concluir que o conjunto de infinitas soluções (formado a partir de infinitos valores de lambda) da segunda proposta engloba as também infinitas soluções da primeira proposta, pois as soluções da primeira proposta nada mais são que as soluções da segunda proposta para os casos que todos os lambdas apresentam o mesmo valor. Assim, vejo que ao utilizarmos lambdas diferentes estamos aumentando o conjunto de possíveis soluções. É como se estivéssemos flexibilizando ainda mais o problema, podendo adotar uma penalização diferente para norma do vetor de pesos de cada classificador separadamente. Podemos pensar que estamos adicionando não apenas um, mas dez hiperparâmetros ao nosso modelo.

Então minha resposta é: sim, adotando um coeficiente de regularização diferente para cada classe pode sim trazer um ganho de desempenho. Claro que isso depende dos dados e do tipo de problema questão, mas o que se sabe é que ao menos as mesmas soluções encontradas com um único lambda serão encontradas também no caso de 10 lambdas, com a vantagem de que temos um conjunto de soluções maiores para explorar no segundo caso.