732A95 INTRODUCTION MACHINE LEARNING

LAB 6: NEURAL NETWORKS AND DEEP LEARNING

JOSE M. PEÑA
IDA, LINKÖPING UNIVERSITY, SWEDEN

INSTRUCTIONS

Each student must submit a report with his/her solutions to the lab. Submission is done via LISAM and before the deadline. The submission file should be named Name_LastName.pdf. The report must be concise but complete. It should include (i) the code implemented or the calls made to existing functions, (ii) the results of such code or calls, and (iii) explanations for (i) and (ii).

PhD students pass the lab if their individual report is of sufficient quality. TDDE01 and 732A95 students pass the lab as follows. The students must discuss their lab solutions in a group. Each group must compile a collaborative report that will be used for presentation at the seminar. The report should clearly state the names of the students that participated in its compilation and a short description of how each student contributed to the report. This report should be submitted via LISAM and before the deadline. The file should be named Group_X.pdf where X is the group number. The collaborative reports are corrected and graded. The individual reports are also checked, but feedback on them will not be given. The students pass the lab if their group report passes the seminar and their individual reports have reasonable quality, otherwise the students must complete their individual reports by correcting the mistakes in them.

Please note that some assignments are mandatory and some are optional, i.e. students must solve the mandatory assignments to pass the lab.

Please use `set.seed(1234567890)` at the assignment to ensure reproducibility.

RESOURCES

The lab is designed to be solved with the R package `neuralnet`. You can find more information on the package's manual and in the following article:

Günther, F. and Fritsch, S. neuralnet: Training of Neural Networs. *The R Journal* 2:30-38, 2010.

1. ASSIGNMENT (**Mandatory**)

Train a neural network to learn the trigonometric sine function. To do so, sample 50 points uniformly at random in the interval $[0, 10]$. Apply the sine function to each point. The resulting pairs are the data available to you. Use 25 of the 50 points for training and the rest for validation. The validation set is used for early stop of the gradient descent. That is, you should use the validation set to detect when to stop the gradient descent and so avoid overfitting. Stop the gradient descent when the partial derivatives of the error function are below a given threshold value. Check the argument `threshold` in the documentation. Consider threshold values $i/1000$ with $i = 1, \ldots, 10$. Initialize the weights of the neural network to random values in the interval $[-1, 1]$. Use a neural network with a single hidden layer of 10 units. **Use the default values for the arguments not mentioned here**. Report the chosen value for the threshold and the final neural network learnt. Feel free to use the following template.

```
library(neuralnet)
set.seed(1234567890)
```

```
Var <- runif(50, 0, 10)
trva <- data.frame(Var, Sin=sin(Var))
tr <- trva[1:25,] # Training
va <- trva[26:50,] # Validation

# Random initializaiton of the weights in the interval [-1, 1]
winit <- # Your code here
for(i in 1:10) {
  nn <- neuralnet(# Your code here)

  # Your code here
}

plot(nn <- neuralnet(# Your code here))

# Plot of the predictions (black dots) and the data (red dots)
plot(prediction(nn)$rep1)
points(trva, col = "red")
```

2. ASSIGNMENT (**Optional**)

Implement the backpropagation algorithm for classification or regression, with the sigmoid or hyperbolic tangent activation function. Use batch or stochastic gradient descent.