MCP261 IE Lab I: Due 11:59 PM March 30, 2022

Exercise 9: Neural networks + nonlinear optimization

Please submit your files in a zipped file. The zipped file should be named as follows: "EntryNum_Name_Ex8.zip". The file should contain a single script containing the answers to both questions below, named as follows: "EntryNum_Name_Ex8.py".

All submissions will be checked for evidence of plagiarism. Students whose submissions are found to have evidence of plagiarism will be subject to, at minimum, losing all marks for the exercise.

1. (5 marks) For the attached data, find the regression neural network architecture (two hidden layers, explore within the range 1-10 for the number of neurons in each hidden layer) that yields the minimum mean absolute percentage error (MAPE).
   Note that the features will have to preprocessed – use MinMaxScaler – prior to fitting the neural net regressor. The first three columns are the features and the last is the regression label.
   Output should be the best neural net architecture (number of neurons in each hidden layer) and the corresponding MAPE value.
   Output format:

   "Best NN architecture: <HL1size>, <HL2size>, <MAPE Score>"

2. (5 marks) For the best neural net architecture, use the fitted neural net as an objective function to find the feature values that maximize the output (the regression label).
   This is a constrained nonlinear optimization problem, with the constraints being bounds – i.e., each feature is constrained to lie in the closed interval [0,1].
   Use one of the global optimizers in the scipy.optimize optimization suite for this purpose. Print out the optimal objective function values and optimal solutions for the best neural net architecture, and also print out the maximum regression label value from the data itself.

   Output format:

   "Optimal objective function value from NN: <objfunvalue>"
   "Maximum regression label value from data: <maxreglabelvalue>"
   "Optimal solution from NN optimization: <feature 1 value>, <feature 2 value>, <feature 3 value>"

   For both questions: Use an 80-20 train/test split for the dataset for both problems, and use the 'tanh' activation function and the 'lbfgs' solver for the neural network. Use a random_state value of 10 wherever applicable. Also use a random number seed of 1234 at the beginning of your code.