Optimization Methods for Data Science, 2023 – 2024 FINAL PROJECT

Alessandro Pannone Veronica Piccialli

13/05/2024

1 Submission Guidelines

Projects must be sent via email to the addresses:

- `veronica.piccialli@uniroma1.it`
- `alessandro.pannone@uniroma1.it`

with the subject line:

`OMDS PROJECT SUBMISSION`

Both members of the team must be included in CC in the email. After the submission, you will receive a confirmation email from the teachers: if you don't receive this email, don't hesitate to get in touch with the teachers to ensure the reception.

The team must attach the Python files (organized as requested in the instructions at the end of the project) **and a typed report** (in PDF format) in a compressed folder named `Surname1_Surname2.zip`, where `Surname1` and `Surname2` are the surnames of the members of the team.

The report must be at most 3 pages for each of the two parts (a total of 6 pages), excluding figures that must be placed at the end of each part. Do not include parts of the Python coding.

2 Evaluation Criteria

Each part is graded up to 30 points (plus one for the bonus question); the final grade is computed by averaging the two parts. In case you answer fully only one of the two Parts (excluding the bonus), you can reach sufficiency (18, not more).

For the evaluation of the Final project, the following criteria will be used:

1. **60% check of the implementation.**
2. **40% quality of the overall project as explained in the report.**

3 Dataset Description

The dataset you will utilize is the **UTK Faces** dataset to train, evaluate and test your models.

The dataset provides a rich resource with over 20,000 labeled images of human faces, each annotated with age (from 1 to 116 years!), gender (Male/Female), and ethnicity labels (5 ethnicity).

In particular, the pictures have been used to train a ResNet for the three tasks and the features extracted by the convolutional backbone are provided as input vectors. The number of features depends on the task. To ensure a more balanced distribution, images of individuals with underrepresented ages in the dataset have been removed.
For the next assignments, three datasets will be provided to you:

- **AGE_REGRESSION.csv**
- **GENDER_CLASSIFICATION.csv**
- **ETHNICITY_CLASSIFICATION.csv**

All three datasets have an header organized as follows:

- the first columns contain the features extracted by the ResNet (these columns are labels as "feat_i"
- the last column is the ground truth (labeled as "gt"). In particular:
  - it is a float number between 0 and 100 for AGE_REGRESSION
  - it is a binary variable for GENDER_CLASSIFICATION
  - it is an integer from 0 to 4 (included) for ETHNICITY_CLASSIFICATION

4 Part 1: Multi-Layer-Perceptron

Dataset: AGE_REGRESSION.csv

In this assignment, you will implement a neural network for regression. The objective is to minimize the regularized L2 loss function. **The network architecture will consist of at least two hidden layers**. **A L2 regularization term will be applied to prevent overfitting**.

The loss function is the following:

- Loss Function:

$$E(\omega, \beta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 + \lambda \sum_{l=1}^{L} ||\omega^{(l)}||^2$$

- Elements:
  - N: Number of training samples
  - y_i: True target value for the i-th sample
  - $\hat{y}_i$: Predicted value for the i-th sample
  - λ: Regularization parameter
  - L: Total number of layers
  - $\omega^{(l)}$: Weight matrix for layer l

You have the flexibility to choose any differentiable activation functions, for instance:

- **Sigmoid**: $\sigma(t) = \frac{1}{1+e^{-t}}$
- **Hyperbolic tangent**: $\tanh(t) = \frac{e^t - e^{-t}}{e^t + e^{-t}}$

The hyperparameters are:

- the number L of layers (min: 2 - max: 4)
- the number N_l of neurons of hidden layers.
- the regularization term. λ.
- the non-linear activation function

To track training/validation/test performances, you can use **Mean Absolute Percentage Error (MAPE)**:

$$\text{MAPE} = \frac{100}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

4.1 Question 1 - Grade up to 30

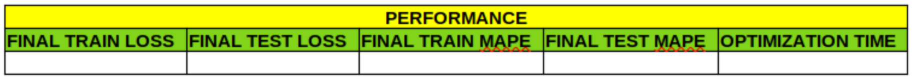
Develop an optimization routine to minimize the $L2$ regularized loss that uses a Python routine for the implementation (`scipy.optimize`) **without utilizing automatic differentiation tools**. **Utilize k-fold cross-validation to select the best hyperparameters.**

4.2 Remarks for the report - Part 1

In the report you must state:

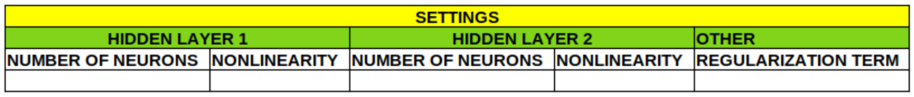
1. the final setting for selected non-linearity, L, N_l, and λ; how and why did you choose them;
2. which optimization routine you used for solving the minimization problem, the setting of its parameters (max number of iterations, etc.) and the returned message in output (successful optimization or others, number of iterations, starting/final value of the objective function) if any;
3. the initial and final values of the regularized error on the training set; the value of the error on the validation set (avg of errors on the k-folds); the final value of the test error.
4. the initial and final MAPE (average of MAPE values obtained with k-fold) on the training set, the final MAPE on validation and test set

In the final report (pdf), it is **MANDATORY** to fill the following tables.



PERFORMANCE				
FINAL TRAIN LOSS	FINAL TEST LOSS	FINAL TRAIN MAPE	FINAL TEST MAPE	OPTIMIZATION TIME

Figure 1: Performance table mlp



SETTINGS				
HIDDEN LAYER 1		HIDDEN LAYER 2		OTHER
NUMBER OF NEURONS	NONLINEARITY	NUMBER OF NEURONS	NONLINEARITY	REGULARIZATION TERM

Figure 2: Settings table mlp

5 Part 2: SVM

Dataset: GENDER_CLASSIFICATION.csv

Consider a nonlinear SVM with kernel $k(\cdot,\cdot)$ and the corresponding nonlinear decision function

$$y(x) = \text{sign}\left(\sum_{i=1}^{L} \lambda_i y_i k(x_i, x) + b\right)$$

where λ, b are obtained as the optimal solution of the dual nonlinear SVM problem.

As kernel function, you can use a Gaussian Kernel $K(x,y) = e^{-\gamma\|x-y\|^2}$ or a Polynomial Kernel $K(x,y) = (x^T y + 1)^p$ where p is a hyper-parameter, which should be set using k-fold cross-validation.

You are requested to train the SVM, namely to both set the values of the hyperparameters C and γ with k-fold cross validation and to find the values of the parameters λ, b with an optimization procedure. For this last task, you need to follow the different optimization procedures described in the following questions.

5.1 Question 2 - grade up to 15

Write a program to find the solution to the SVM dual quadratic problem. Apply **k-fold cross validation** for identifying the values of the hyperparameters C and γ. To find the solution of the SVM dual quadratic problem, you should use a specific method for convex optimization, such as **CVXOPT**.

5.2 Question 3 - grade up to 15

Fix the dimension of the subproblem to $q = 2$ and implement the most violating pair (MVP) decomposition method, which uses the analytic solution of the subproblems.

5.3 BONUS Question 4 - +1 point

Dataset: ETHNICITY_CLASSIFICATION.csv

Consider a three classes problem with three out of the five ethnicities in the dataset. **Implement an SVM strategy for multiclass classification** (either one against one or one against all). For this task, is not mandatory to use k-fold cross validation to tune SVMs hyperparameters.

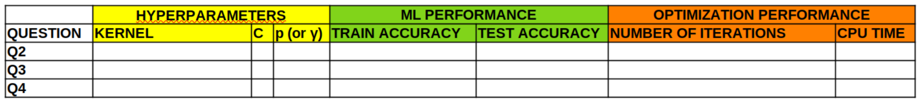
5.4 Remarks for the report - Part 2

- Only for Question 2: the final setting for the hyperparameter C (upper bound of the constraints of the dual problem) and of the hyperparameter of the kernel chosen; how you have chosen them and if you could identify values that highlight over/underfitting;
- Only for Question 2: which optimization routine you use for solving the quadratic minimization problem and the setting of its parameters, if any (write DEFAULT if you have not changed them);
- For Question 2 and 3:

- Machine learning performances: write the value of the accuracy on the training and test set; (for question 2, write also the value of the validation accuracy).
- Optimization performance: report the initial and final value of the objective function of the dual problem and the number of iterations, either as it is returned by the optimization routine used or evaluated by yourselves.

- For question 3:
  - Machine learning performances: write the value of the accuracy on the training and test set;

The comparison among all the implemented methods - in terms of accuracy in learning and computational effort in training - must be gathered into a final table.



	HYPERPARAMETERS			ML PERFORMANCE		OPTIMIZATION PERFORMANCE	
QUESTION	KERNEL	C	p (or γ)	TRAIN ACCURACY	TEST ACCURACY	NUMBER OF ITERATIONS	CPU TIME
Q2							
Q3							
Q4							

Figure 3: Results table svm

6 Instructions for Python code

You are allowed to organize the code as you prefer BUT for each question, you have to create a different folder where you must provide the files according to the following instructions.

6.1 Part 1 - MLP

- For each question, you have to provide a file called `Functions_ij_surname1_surname2.py`. This file should only include all the classes, functions, and libraries you used for solving the specific question i in part j.
- A file called `run_1i_surname1_surname2.ipynb`. This file should include the import of all the functions you need, but it has to print ONLY:
  1. Number of layers L chosen
  2. Number of neurons N chosen
  3. Value of λ chosen
  4. Values of other hyperparameters (if any)
  5. Optimization solver chosen (e.g. L-BFGS, CG, NTC, Nelder-Mead...)
  6. Number of iterations

7. Time for optimizing the network (from when the solver is called until it stops)
8. Training Error (defined as in question (1), without regularization term)
9. Test Error (defined as above but on the test set)

- If you want, you can put all your functions, objects etc. directly in python notebook (if you don't want to use Functions.py file) BEFORE THE CODE

6.2 Part 2 - SVM

- For each question, you have to provide a file called `Functions_ij_surname1_surname2.py`. This file should only include all the classes, functions, and libraries you used for solving the specific question i in part j.
- A file called `run_2i_surname1_surname2.ipynb`. As in part 1, it has to print ONLY:
  - Setting values of the hyperparameters
  - Classification rate on the training set (% instances correctly classified)
  - Classification rate on the test set (% instances correctly classified)
  - The confusion matrix
  - Time necessary for the optimization procedure
  - Number of optimization iterations
  - Final difference between $m(\lambda)$ and $M(\lambda)$ (only for question 4)
  - Final value of the objective function of the dual SVM problem
- If you want, you can put all your functions, objects etc. directly in python notebook (if you don't want to use Functions.py file) BEFORE THE CODE

General Remarks concerning Python code

1. Each question will be graded by averaging the Performance score (25%), the Efficiency score (25%), and the Mathematical Correctness score (50%).
2. In **Question 1**, to get the maximum efficiency score, **DO NOT WRITE FOR LOOPS IN FORWARD AND BACK PROPAGATION, use numpy vectorization only**
3. There is no competition with other students. Your grade does not depend on other students' performances.

4. You are warmly recommended to write clean, commented, and easily readable code. Not respecting general assignments (for example, printing different values or not printing requested outputs) will result in penalization.

5. Your code will not be debugged and will not be modified in ANY case after the submission. If the code does not run, the resulting score will be zero. The code will be executed in Google Colab, so you are warmly recommended to check that your code can successfully execute on this platform. The provided colab notebook will be executed from the first to the last cell, be careful on where you define your functions!

6. It is welcomed to take inspiration from other students, as well as from open-source contents online, but ”copy and paste” projects will not be considered valid. Be aware that your code will be cross-checked with online materials, as well as with other students’ code.

7. The use of any library automatically building neural models (e.g., Pytorch, TensorFlow, sklearn.neuralnetwork, etc.) is FORBIDDEN and automatically invalidates the assignment.