Optimization Methods for Machine Learning - Fall 2021

PROJECT # 2
Support Vector Machines

Laura Palagi & Giulia Di Teodoro

Posted on November 21, 2021 - due date December 20, 2021

Instructions

The project will be done in groups formed by 2 to 4 people: each group must hand in their own answers. We will be assuming that, as participants in a graduate course, every single student will be taking the responsibility to make sure his/her personal understanding of the solution to any work arising from such collaboration.

Submission

Projects must be uploaded using a google form

https://forms.gle/2kcsHX3QdohbnsVJ7

Each team leader is in charge for uploading the files.

The team must upload the phyton files organized as requested in the instructions at the end of the project, in a compressed folder "GroupNumber"_"GroupName"_code.zip.
A typed report in English and in pdf format must be uploaded too. **The report must be of at most 4 pages excluded figures that must be put at the end. Do not include part of the phyton coding.** The report must be named "GroupNumber"_"GroupName"_report.pdf

Evaluation criteria

The grade is "Italian style" namely in the range [0,30], being 18 the minimum degree to pass the exam. Project is due at latest at midnight on the due date. For late submissions, the score will be decreased. It is worth 85% for the next 48 hours. It is worth 70% from 48 to 120 hours after the due date. It is worth 50% credit after 120 hours delay.
The second project accounts for 35% of the total vote of the exam.
For the evaluation of the second 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.

In this project you will implement different optimization methods for training Support Vector Machines (supervised learning) in order to solve a classification problem.

Data sets.
You are asked to build a classifier that distinguishes between scan images of capital letters of English alphabet. It is a good database for trying learning techniques on real-world data while spending minimal efforts on preprocessing and formatting. The database is made by a large number of black-and-white rectangular pixel referred to the 26 capital letters in the English alphabet. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15. The columns of the dataset are referred to the following attributes:

1. x-box: horizontal position of box (integer)
2. y-box: vertical position of box (integer)
3. width: width of box (integer)
4. high: height of box (integer)
5. onpix: total # on pixels (integer)
6. x-bar: mean x of on pixels in box (integer)
7. y-bar: mean y of on pixels in box (integer)
8. x2bar: mean x variance (integer)
9. y2bar: mean y variance (integer)
10. xybar: mean x y correlation (integer)
11. x2ybr: mean of x * x * y (integer)
12. xy2br: mean of x * y * y (integer)
13. x-ege: mean edge count left to right (integer)
14. xegvy: correlation of x-ege with y (integer)
15. y-ege: mean edge count bottom to top (integer)
16. yegvx: correlation of y-ege with x (integer)
17. letter: capital letter. This is the column of the output y.

It is necessary to scale the data before the optimization.

You will be given a full dataset and you will extract the target set made up of images belonging to two (Question 1,2,3) or three (bonus question) classes.

For the Questions 1, 2 and 3, the target set will be made of images of the character O and Q (identified by the labels O and Q of the "letter" column, in the target set, respectively). In the bonus question, there will also be images of D (label "D"). You are not given a test set. You can obtain it by randomly splitting the target set into a training set and a test set with a percentage of 80 %, 20%. Use one of your matricola numbers (or another fixed number) as a seed for the random split. A $k-$fold cross validation can be used to set the hyperparameters. Please note that the teacher has defined a test set that might be used to check the accuracy of your models.

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

$$y(x) = sign\left(\sum_{p=1}^{P} \alpha_p y^p k(x^p, x) + b\right)$$

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

As kernel function you may choose either a RBF kernel

$$k(x,y) = e^{-\gamma\|x-y\|^2}$$

where $\gamma > 0$ is an hyper parameter, or a polynomial kernel

$$k(x,y) = (x^T y + 1)^\gamma$$

with hyper parameter $\gamma \geq 1$.

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

Question 1. (max score up 24) Write a program to find the solution of the SVM dual quadratic problem. Apply any procedure for identifying the values of the hyperparameters C and γ. To find the solution of the SVM dual quadratic problem, you can use any routine that exploits the gradient of the objective function. Note that using *specific* method for Quadratic Programming problems would be preferable and it will lead to better optimization performances.

Question 2. (max score up to 27) Use the same kernel and hyperparameters of Question 1. Write a program which implements a decomposition method for the dual quadratic problem with any **even** value $\mathbf{q} \geq \mathbf{4}$. You must define the selection rule of the working set, construct the subproblem at each iteration and use a standard algorithm for its solution. You can use any routine that exploits the gradient of the objective function. However the use of specific method for quadratic programming would be preferable. You must implement in the outer optimization loop a stopping criterion based on the optimality conditions.

Question 3. (max score up to 30) 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.

Question 4. Additional bonus exercise. (1 point bonus)
Consider a three classes problem with the classes of letters Q, O, D. Implement a SVM strategy for multiclass classification (either one against one or one against all).
In the report you must state:

- the chosen kernel;
- Only for Question 1: the final setting for the hyper parameter C (upper bound of the constraints of the dual problem) and of the hyper parameter of the kernel chosen; how you have chosen them and if you could identify values that highlight over/under fitting;
- Only for Question 1 and Question 2: which optimization routine you use for solving the quadratic minimization problem and the setting of its parameters, if any;
- For each Question:
    - machine learning performances: report the value of the accuracy on the training and test set; (for question 1, report also the value of the validation accuracy, if computed).
    - optimization performance: report the initial and final value of the objective function of the dual problem, the number of iterations, the number of function evaluations, the violation of the KKT condition, either as it is returned by the optimization routine used or evaluated by yourselves.

The comparison among all the implemented methods - in term of accuracy in learning and computational effort in training - must be gathered into a final table (an example below)

	hyperparameters			ML performance		optimization performance			
question	C	γ	q	Training accuracy	Test accuracy	number its	number fun evals	KKT viol	cpu time
Q1									
Q2									
Q3									
Q4									

Instructions for python code

You are allowed to organize the code as you prefer **BUT** for each question $i = \{1,2,3,4\}$ you have to create a different folder where you must provide two files:

- A file called **run_i_GroupName.py** (e.g. run_1_Example.py,). This file will be the only one executed in phase of verification of the work done. It can include all the classes, import all the functions and libraries you used for solving the specific question *i* but 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
  - number of optimization iterations
  - difference between $m(\alpha)$ and M(α)

  Furthermore, in question 2 you must print also:
  - value of q

  Please, in order to maintain the code as clean/clear as possible, do not define functions inside the run_i_GroupName.py file, but make it call functions defined in other files.
- A file with the complete code you have written that includes all the functions that are called from the **run_i_GroupName.py file** and all the procedures that have been used to select the hyperparameters.

  Further instructions for the python code might be provided in the near future.