FTML practical session 3: 2023/03/23

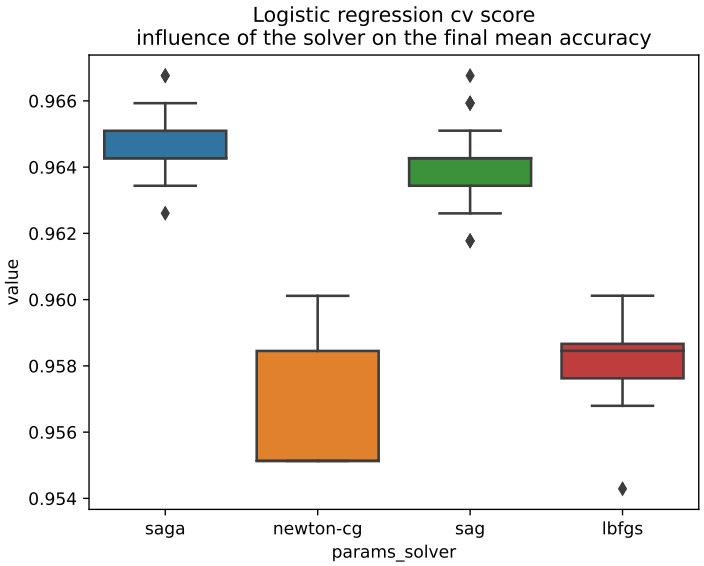

TABLE DES MATIÈRES

INTRODUCTION

In the previous session, we encountered our first hyperparameter (HP), namely the regularisation constant λ in a Ridge regression problem. We saw that in that situation, there are some optimal values for λ, that lead to an optimal generalization error. But unfortunately, most of the time, we do not have access to these optimal values, by lack of information of the data.

The goal of this session is to study different methods used to tune hyperparameters in a machine learning problem. Most machine learning problems, and many optimization problems, do have some hyperparameters (often several of them), that must be tuned from the data. This session is more oriented towards implementation and manipulating ML libraries, hence we will use some algorithms like SVC, trees, or logistic regression as black-boxes for now, but you can read their documentation.

The choice of HPs is part of the topic of **model selection**. There are two important aspects of HP tuning that we will discuss :

— how to select HP set values

— how to evaluate the quality of a set of HPs

https://scikit-learn.org/stable/model_selection.html

1 VALIDATION TESTS AND CROSS VALIDATION

1.1 Validation sets

1.1.1 *Definitions*

A basic approach for comparing hyperparameter sets is to separate the dataset into 3 parts :

— a train set : e.g. given a set of HPs, used to solve the optimization problem if we do empirical risk minimization.

— a validation set : used to compute the score of the predictor that was learned with this set of HPs.

— a test set : after trying all sets of HPs, used to compute the score of the predictor that had the best validation score.

The test set must be used only once : it gives an unbiased estimator of the generalization error, as opposed to the validation error, that is **not** unbiased.

1.1.2 *Issues*

There are several possible issues with this approach, the main one being that it is possible that the HPs "overfit the validation set". This means that it is possible that a given set of HPs lead to a good score on a specific choice of the validation set, but does not generalize as well. As the choice of the vlaidation set is usually arbitraty and random, this might lead to a large variance of the predictor obtained, and a worse generalization error.

1.1.3 *Simulation*

The previous issue is not always a big one. Usually, it happens when predictors with large variance are used, and / or with datasets that are too small to contain enough relevant statistical information about the data. We will illustrate this on a classification problem.

Perform a simulation with the train / validation / test split in order to perform a hyperparameter tuning for which the test score is significantly, and robustly worse than the validation score.

Suggestions :

— use for instance the digits dataset or the wine from scikit, and extract only a random subset of this dataset ; e.g. 300 points or less.
https://scikit-learn.org/stable/datasets/toy_dataset.html

— use **train_test_split()** from scikit to perform the splits.
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

— use **ParameterGrid()** to build a **grid** that you can iterate over to compare HPs.
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.ParameterGrid.html

— use **DecisionTreeClassifier()** or **SVC()** (support vector classifier) from scikit and choose some HPs from these classes to tune the tree learned.
https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier

You will see that the validation score is not always worse than the test score, depending on the HP grid, and of the size of the dataset !

1.2 Cross validation

Cross-validation (CV) is another approach to test the quality of a set of HPs, that is usually more robust to statistical noise. The dataset is split only in a train set and a test set, but the train set is used differently. It is itsself splitted multiple times in **folds**, and an **average** of errors on test folds is computed. The main issue with CV is that it is longer : more opitmizations have to be performed.

https://scikit-learn.org/stable/modules/cross_validation.html

https://en.wikipedia.org/wiki/Cross-validation_(statistics)

1.2.1 *Simulation*

Run cross validation on the same problem as before in order to obtain a better test score than with the classical train / validation / test approach. You will notice that cross validation itsself has parameters, such as the number of splits in the dataset.

— https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html

— https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

— In scikit-learn, many estimators have wrappers in order to directly tune hyperparameters with cross validation.
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.RidgeCV.html

2 HYPERPARAMETER TUNING METHODS

2.1 Grid search and random search

The approach that we just saw is the most basic one : choose a list of values for each and iterate through a grid containing all possible combinations, this is called **grid search**.

https://scikit-learn.org/stable/modules/grid_search.html#

See also random search, and successive halving.

2.1.1 *Issues*

Grid search might be suboptimal in the sense that all hyperparameter combinations will be tried, although many of them might be irrelevant. Also, the size of the grid is exponential in the number of parameters.

2.1.2 *Question*

Why can we usually not run a gradient-based optimization algorithm to look for better HP values ?

2.2 Bayesian optimization

More modern approaches use smarter optimization methods to tune the hyperparameters. For instance, **Bayesian optimization**.

https://en.wikipedia.org/wiki/Bayesian_optimization

2.2.1 *Frameworks*

Several frameworks exist to perform Bayesian optimization over HPs values, and automate the search.

skopt https://scikit-optimize.github.io/stable/

optuna https://optuna.org/

With these methods, we can specify ranges for scalar HP values, instead of grids of values.

Perform HP optimization with optuna over the same problem as before, or over a different one. You can still use SVC, trees, or investigate for instance logistic regression.

Optuna creates "studies", in which HP sets are tested in a sequence. A set of HPs is tested in a "trial".

https://optuna.readthedocs.io/en/stable/reference/study.html

Some interesting optuna aspects to consider :

— some trials can be "pruned" : they are considered as not interesin by optuna, and are discarded before the optimization is finished.

— it is possible to start a study, stop the program, and resume the study later, with the previous trials still in the study database.

— you can analyze the results of a study by processing the study object with other libraries like pandas, seaborn, matplotlib, etc.

2.2.2 *Optuna dashboard*

With optuna, you can display and analyze the influence of the HP with a dashboard that opens in your browser.

`https://github.com/optuna/optuna-dashboard`

Implement the dashboard in your simulation and use it to analyze HP optimization in a problem of your choice.