Advanced Machine Learning: Assignment 5

Raffaele Anselmo[*][†]

Abstract

The task of Assignment 5 is Hyperparameter Optimization of a Neural Network, with the aim to maximize its Accuracy on 10 folds Cross Validation.

1 Introduction

Hyperparameter Optimization is the task of AutoML that aims to find out the best subset of hyperparameters of a given algorithm.

The Task consists of 2 step: in the first one, a neural network with 2 hidden layer (4 neurons in the first hidden layer and 2 in the second one) has to be optimized with respect of learning rate (range 0.01-0.1) and momentum (0.1-0.9). Two experiments will be conducted: one with Expected Input as acquisition function and one with Lower Confidence Bound. The results will be compared with Grid Search and Random Search; in the second step, along with the two previous hyperparameters, the number of units in the two hidden layers have to be optimized in the range 1-5.

The dataset is named "fertility" and consists of 100 observation, 9 explanatory variables and a binary unbalanced target class (12 positive records and 88 negative ones). It is available on OpenML [1]. For the optimization purposes SMAC3 package [2] will be deployed, while the Neural Network will be defined with the help of scikit-learn library [3].

2 Step 1

First, Configuration Space and Scenario are defined. In particular 2 Uniform Float Hyperparameters, Learning Rate and Momentum, will be added without any conditional hypotheses. Their range is defined into lr/momentum _lower and _upper objects and is added in the cs Configuration Space. Then, the same initial random design is set for the two experiments and it consists of 5 runs. The remaining 20 runs will be up to the acquisition function used. To complete

[*]Email: `r.anselmo@campus.unimib.it`
[†]Matricola: 846842

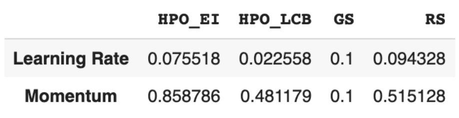
the scenario, run_obj is set to “quality” and deterministic “True”, so a single value will be returned for every run.

For the evaluation two function will be defined: the first is plot_bestseen, that plots the best seen evolution of an optimization experiment, and the second is CR, that consists of the same algorithm used for optimization and a predictor. It will return the Classification Report.

Once created configuration space, scenario and results dictionary (empty at the moment) and defined the utility functions, everything is ready to pass the objects to SMAC.

The neural network is defined in nn_from_cfg: it consists of 2 hidden layers with 4 neurons in the first layer and 2 in the second one. Its hyperparameters are the default ones (activation=‘relu’, solver=‘adam’). Before fitting the model, a Smote operation will be used to deal with the unbalanced class and a standardization will be performed. The function will return 1-(accuracy mean in 10 folds cross validation) because SMAC need a function to be minimized, and task is about maximizing the accuracy.

In run_smac function the optimization will be performed. It requires the acquisition function as Input and will utilize a Bayesian Optimization with Random Forest surrogate function. The best configuration and best seen values will be added to the results dictionary. Once performed the optimizations with the two different acquisition functions, the last part of the task consists on a comparison with Grid Search and Random Search. The best configurations founded with the four hyperoptimization techniques are summarized in Figure 1:



	HPO_EI	HPO_LCB	GS	RS
Learning Rate	0.075518	0.022558	0.1	0.094328
Momentum	0.858786	0.481179	0.1	0.515128

Figure 1: Step 1 incumbent configurations

The results obtained with SMAC are different both for learning rate and Momentum. This is explained by the different points used between the two acquisition functions, that led to different (local) minima. For what regards grid search and random search, their results are the same in learning rate, but different in momentum. In general, the configurations result different for all the optimization techniques used.

Once found the best configurations, they will be used for the classification. The results are very similar and they are summarized in four classification reports (Fig.2), while the evolution of the “best seen” is plotted in Fig. 3.

```
              precision    recall  f1-score   support

         0.0       0.14      0.33      0.20        12
         1.0       0.89      0.73      0.80        88

    accuracy                           0.68       100
   macro avg       0.52      0.53      0.50       100
weighted avg       0.80      0.68      0.73       100
```

(a) HPO - Expected Input

```
              precision    recall  f1-score   support

         0.0       0.24      0.42      0.30        12
         1.0       0.91      0.82      0.86        88

    accuracy                           0.77       100
   macro avg       0.57      0.62      0.58       100
weighted avg       0.83      0.77      0.80       100
```

(b) HPO - Lower Confidence Bound

```
              precision    recall  f1-score   support

         0.0       0.16      0.33      0.22        12
         1.0       0.89      0.76      0.82        88

    accuracy                           0.71       100
   macro avg       0.53      0.55      0.52       100
weighted avg       0.81      0.71      0.75       100
```

(c) Grid Search

```
              precision    recall  f1-score   support

         0.0       0.16      0.33      0.22        12
         1.0       0.89      0.76      0.82        88

    accuracy                           0.71       100
   macro avg       0.53      0.55      0.52       100
weighted avg       0.81      0.71      0.75       100
```

(d) Random Search

Figure 2: Step 1 Classification Reports

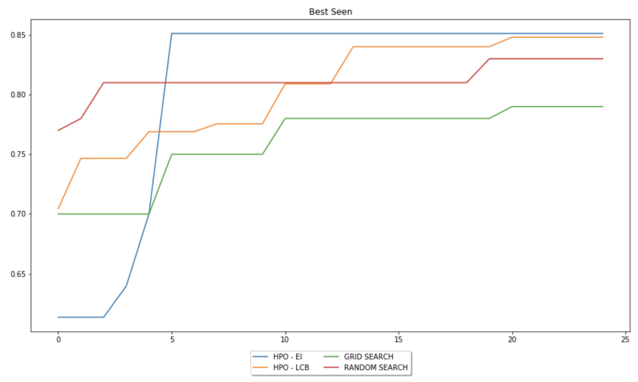


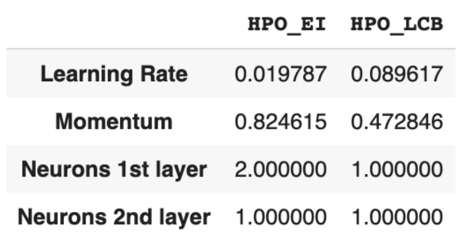
Figure 3: Step 1 best seen evolution

3 Step 2

In this step, along with the two previous hyperparameters, Learning Rate and Momentum, also the number of neurons in each of the two hidden layer will be optimized. This search over the neural network architecture evolves our task in a SMBO (Sequential Model Based Optimization).

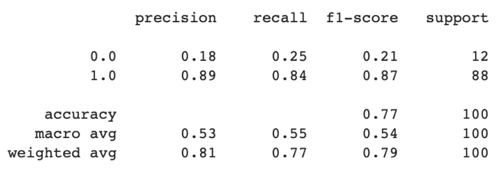
The procedure followed in this step is the same as the previous one, with the Configuration Space and Scenario definition (with the addition of the two Uniform Integer Hyperparameters, neurons_1_layer and neurons_2_layer), followed by the utility functions definition and finally by the SMAC optimization. The budget is larger and consists of 10 initial runs with the same random design for the two experiments and 100 further runs.

The configurations are, again, different between the two experiments (Fig.4), while the results are more balanced for the first experiment. Actually, with the second configuration, no one record has been predicted with the rare class. This issue might have been fixed using other objective metrics (like F1-measure), more appropriate for the unbalanced class case, but this is outside the task. (Fig.5):



	HPO_EI	HPO_LCB
Learning Rate	0.019787	0.089617
Momentum	0.824615	0.472846
Neurons 1st layer	2.000000	1.000000
Neurons 2nd layer	1.000000	1.000000

Figure 4: Step 2 incumbent configurations

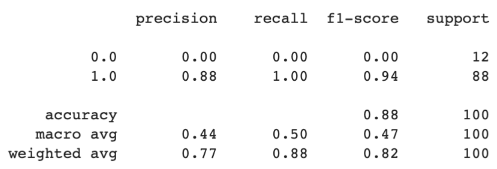


```
              precision    recall  f1-score   support

         0.0       0.18      0.25      0.21        12
         1.0       0.89      0.84      0.87        88

    accuracy                           0.77       100
   macro avg       0.53      0.55      0.54       100
weighted avg       0.81      0.77      0.79       100
```

(a) HPO - Expected Input



```
              precision    recall  f1-score   support

         0.0       0.00      0.00      0.00        12
         1.0       0.88      1.00      0.94        88

    accuracy                           0.88       100
   macro avg       0.44      0.50      0.47       100
weighted avg       0.77      0.88      0.82       100
```

(b) HPO - Lower Confidence Bound

Figure 5: Step 2 Classification Reports

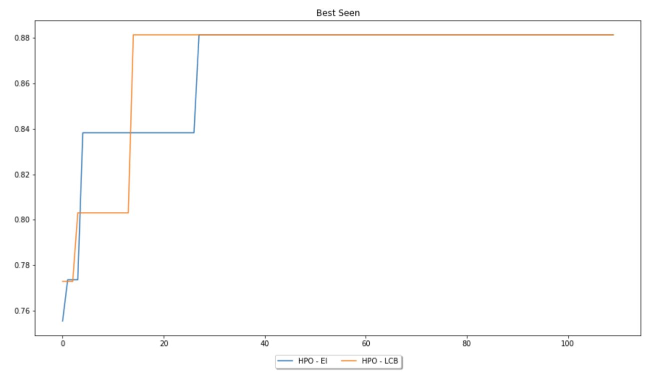


Figure 6: Step 2 best seen evolution

4 Conclusion

Although using the Smote operator an oversampling has been performed on minority class, the classifier does not perform well on predicting the rare class.

Further works should use other target metrics, such as F1-measure. The performances are very similar for the classifiers trained with the different configuration obtained from the different hyperoptimization techniques and SMBOs.

For what regards the configurations, both in step 1 and in step 2, they are very different for the several hyperoptimization techniques and acquisition functions used. This confirms the multi-extremal form of the objective function to be optimized with respect of the hyperparameters, with the presence of many (local) minima, that could be very far from each other.

In conclusion, while in this small dataset it cannot be appreciated the relevance of hyperoptimization and SMBO, it has been useful to exploit the multi-extremal form of the hyperoptimization function and the role of the acquisition function in the process.

References

[1] Fertility dataset. https://www.openml.org/d/1473.

[2] Marius Lindauer, Katharina Eggensperger, Matthias Feurer, Stefan Falkner, André Biedenkapp, and Frank Hutter. Smac v3: Algorithm configuration in python. https://github.com/automl/SMAC3, 2017.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.