Iran Izadyariaghmirani

Practical Machine Learning Homework 4: ML Hyperparameter Optimization

(I) Introduction

Hyperparameter optimization plays a pivotal role in the field of machine learning. For any machine learning model, these parameters must be fine-tuned to achieve optimal performance. In this exercise, we explore various hyperparameter optimization techniques applied to predictive models using the "*wine quality*" dataset. The objective is first to find the model with the best predictive accuracy and, secondly, to understand the impact of hyperparameter tuning on different machine learning algorithms.

(II) Dataset description

The dataset utilized in this exercise is the "winequality-red" dataset, which includes data on various features of wines, such as fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, and alcohol. The dataset comprises a total of 1,599 rows and eleven features. The target variable, "quality," rates the wine on a scale from zero to ten (0-10). There are no missing values in the dataset. The dataset was preprocessed to adjust features' scales and distributions, applying a logarithmic transformation followed by standardization to normalize the data.

(III) Methodology

☐ ***Programming languages and libraries:***

Our investigation was conducted in Python, utilizing Pandas for data management, Scikit-learn for implementing machine learning algorithms, and Scikit-optimize for efficient hyperparameter optimization through Bayesian optimization techniques.

☐ ***Data preparation and splitting:***

The dataset was first preprocessed to ensure optimal conditions for model training. This preprocessing included a log transformation to address skewness in the feature distributions, followed by standard scaling to normalize feature scales. The dataset was then divided into training and test sets, with 80% allocated for training and the remaining 20% for testing. This split was stratified by the wine quality rating to maintain a balanced representation across both sets and ensure that the model is trained and evaluated on data that accurately reflects the overall distribution of the dataset.

☐ ***Machine learning algorithms and hyperparameters***

- *Selected algorithms:* We focused on three classification algorithms, including "Random Forest", "Support Vector Machine (SVC)", and "Logistic Regression", to cover a broad range of learning strategies from ensemble methods to linear models.
- *Hyperparameter space*: The selection of hyperparameters and their corresponding search spaces was a critical step in our hyperparameter optimization process. To systematically explore and identify the best configurations for our models, we defined specific ranges for each hyperparameter based on prior research, practical considerations, and the models'

documentation. Below is a detailed overview of the hyperparameter spaces for the Random Forest Classifier, Support Vector Machine (SVC), and Logistic Regression:

1. *Random Forest Classifier*
    - n_estimators**:** Number of trees in the forest. The search space ranged from 100 to 500, allowing for a broad exploration of moderate and more complex forest sizes.
    - max_depth**:** Maximum depth of the trees. We considered depths from 10 to 50 to evaluate the impact of tree complexity on performance.
    - min_samples_split*:* Minimum number of samples required to split an internal node, with a search space from 2 to 10. This range allows for the investigation of both highly granular and more generalized splits.
2. *Support Vector Machine (SVM)*
    - C (Regularization parameter): A log-uniform distribution range from 0.1 to 10 was chosen to reflect a wide spectrum of regularization strengths from very weak to very strong. Parameter *"C"* controls how strongly the model prevents misclassifying training examples.
    - Gamma (Kernel coefficient): This parameter defines the influence of individual training samples on the decision boundary. When gamma is high, the 'reach' of the data points is more limited, which means the decision boundary will depend highly on the data points closest to it. Conversely, a low gamma spreads the influence of the data points wider, leading to a smoother decision boundary. The gamma selection is as ['scale', 'auto'], which means it can automatically select the gamma value as either 'scale' (1 / (number of features * variance of X)) or 'auto' (1 / number of features).
3. *Logistic Regression*
    - C (Inverse of regularization strength): Similar to SVC, this parameter, varying log-uniformly between 0.1 and 10, controls the strength of the regularization applied to the model, which can help to avoid overfitting by penalizing larger values of the coefficients.

☐ ***Hyperparameter optimization method and evaluation***

In the tuning phase, we executed a systematic exploration of hyperparameters for multiple classifiers. Each classifier, encapsulated within a pipeline that included both preprocessing and the classifier itself, was subjected to Bayesian optimization. The Bayesian optimization was implemented through the 'BayesSearchCV' interface with a random_state of 42. The ML models were performed at ten (10) iterations of hyperparameter adjustments within a 5-fold cross-validation framework while targeting maximized accuracy. The Bayesian optimization process intelligently navigates the hyperparameter space and efficiently identifies promising regions by building a probabilistic model of the function mapping hyperparameters to model performance.

Model performance was rigorously evaluated using accuracy as the metric, applied through a 5-fold cross-validation approach on the training data to ensure a reliable assessment of the model's predictive capabilities. This cross-validation strategy, coupled with a separate test set for final evaluation, was designed to mitigate overfitting and provide a trustworthy estimate of model performance on unseen data. The key results from the Bayesian optimization included metrics such as the mean CV accuracy (computed as the best score achieved during the tuning process), the accuracy on the unseen test data, and the optimal set of hyperparameters.

(IV)Results

The mean CV accuracies obtained from the ML classifiers before and after optimizing the hyperparameters are listed in Table 1. Moreover, better comparative representations of the original (with default hyperparameters) versus the tuned models for the training and test data are depicted in Figures 1 and 2, respectively. The data in Table 1 demonstrates that, for the default models, the Random Forest has shown better performance and a higher mean cross-validation accuracy compared to the SVM and Logistic Regression models for both training and test data. This finding highlights the superiority of this model compared to the two other approaches.

Table 1- Results of the ML classifiers before and after optimizing the hyperparameters with random state of 42

Classifier	Mean accuracy (mean CV accuracy for training)				Improvement on test accuracy (%)
	Default – training data	Tuned – training data	Default – test data	Tuned – test data	
Random Forest	0.6779	0.6873	0.6469	0.6656	+0.0188 (2.9)
Logistic Regression	0.6044	0.6076	0.5719	0.5625	-0.0094 (-1.6)
SVM	0.5943	0.6216	0.5750	0.6094	+0.0344 (6.0)

Note*: Mean CV accuracy refers to the mean cross-validation accuracy obtained during the ML process.*

Furthermore, the results in Figure 1 illustrate that the mean CV accuracies during training have been consistently improved for all models after tuning the hyperparameters, with the best improvement to be observed for the SVM model. This finding emphasizes the enhanced ability of the models to learn from the training data with optimized hyperparameters.

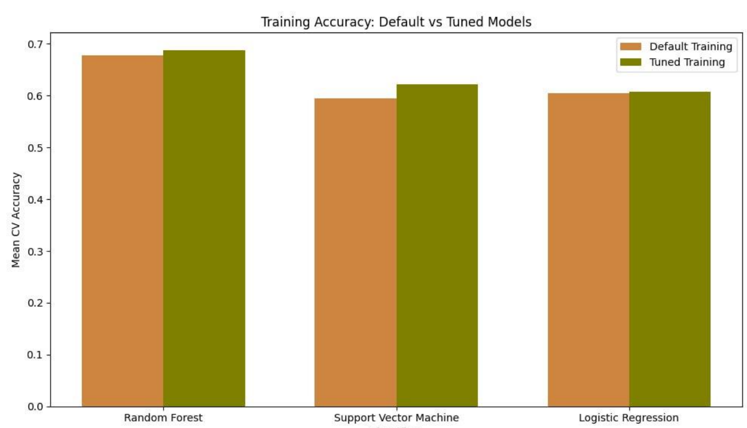

Figure 1 – The mean CV accuracies-training-data of various classifiers before and after the hyperparameter optimization

The model performance for test data in Figure 2 highlights that the predictions of unseen data using Random Forest and SVM models have notably benefited from the hyperparameter optimization on training data. The mean test CV accuracies of these models have been improved by 2.9 % and 6.0%, respectively. The SVM model had the least performance before tuning, whereas the hyperparameter optimization made it superior to the Logistic regression for both the training and testing phases. These observations indicate the efficacy of Bayesian optimization in refining model configurations to better capture the underlying patterns in the wine quality dataset.

For the Logistic Regression method, although the training accuracy was improved using the hyperparameter optimization approach, the results on the test accuracy slightly declined compared to the default model. This marginal post-tuning accuracy reduction suggests that the selected hyperparameter space might not have been optimal or that the model was inherently less suitable for this specific dataset. It is noteworthy that extending the iterations from 10 to 100 produced no substantial improvement in accuracy.

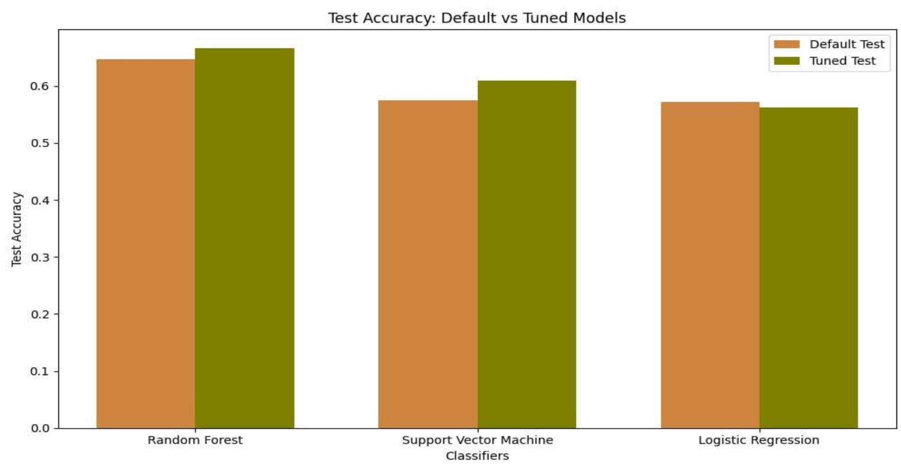

Figure 2 – The mean CV accuracies-test-data of various classifiers before and after the hyperparameter optimization

(V) Further investigation of ML hyperparameter optimization

As we discussed in the results, the hyperparameter tuning for the "Logistic Regression" method resulted in a negative effect on the accuracy of test data. This result was repeated even after changing the regularization strength 'C'. Therefore, we focused on two other stages that are discussed as follows.

- ***Data characteristics and preprocessing***

Logistic regression is sensitive to data scaling and distribution. Initially, the pre-processing of the data was conducted by applying both logarithmic transformation and standard scaling to the same features, which might have introduced complexities that are unfavorable for logistic regression. This is despite the fact that these steps could have been helpful for more complex models such as random forests and SVM. To further investigate this hypothesis, two additional scenarios were performed on the logistic regression model: by applying no preprocessing, and another one by using only the scaling process. In both cases, no positive improvements were observed for thee test data, and hence, it was decided to maintain the pre-processing stage as it was.

- **Changing the random state**

Altering the random state in machine learning models can lead to different results, especially regarding performance improvements after hyperparameter tuning. In this case, the random state was changed from 42 to 24. The results of the hyperparameter tuning with random states are summarized in Table 2.

Table 2- Results of the ML classifiers before and after optimizing the hyperparameters with random state of 24

Classifier	Mean accuracy (mean CV accuracy for training)				Improvement on test accuracy (%)
	Default – training data	Tuned – training data	Default – test data	Tuned – test data	
Random Forest	0.6802	0.6927	0.6469	0.6500	+0.0031 (0.48)
Logistic Regression	0.5934	0.6177	0.5781	0.5813	+0.0031 (0.48)
SVM	0.5943	0.6177	0.5563	0.6031	+0.0469 (8.4)

The results indicate that, when a random state of 24 was used, the hyperparameter tuning procedure resulted in improvements of the accuracy for both training and test data under the logistic regression classification method. Logistic regression, as a linear model, may inherently be less capable of capturing complex patterns in the data compared to non-linear models such as SVMs or random forests. This limitation may become more apparent during the hyperparameter tuning process and hence a smaller random state is recommended in this case.
It is noteworthy that, for the SVM and Random Forest methods, the accuracies of the test data were also improved when using a random state of 24. Nevertheless, the final results were inferior compared to those obtained with a random state of 42.

(VI) Conclusion

This study highlights the importance and effectiveness of hyperparameter optimization in improving the predictive performance of machine learning models. Through a structured approach utilizing Bayesian optimization, we were able to achieve notable improvements for the accuracy (both test and training) of Random Forest and SVM methods on the Wine Quality dataset. The test accuracy of the logistic regression model was only improved when a low value of random state was selected. These findings demonstrated the potential of hyperparameter tuning in enhancing model accuracy and the value of selecting appropriate optimization methods and hyperparameter spaces for different machine-learning algorithms.

(VII) References

1) Domingos, Pedro. "A few useful things to know about machine learning." Communications of the ACM 55.10 (2012): 78-87.
2) Kim, Dohyung, Jahwan Koo, and Ung-Mo Kim. "A Survey on Automated Machine Learning: Problems, Methods and Frameworks." International Conference on Human-Computer Interaction. Cham: Springer International Publishing, 2022.
3) A Comprehensive Guide on Hyperparameter Tuning and its Techniques (https: analyticsvidhya.com)