Iran Izadyariaghmirani

Practical Machine Learning Homework 5: Pipeline Optimization

(I) Introduction

The core problem addressed in this report involves optimizing the complete machine learning (ML) pipeline rather than individual components, for the '*winequality-red'* dataset. Our focus is on the systematic tweaking of hyperparameters across various stages of preprocessing and model selection, aiming not just for performance but for a deep understanding of the influence and interaction of each component within the pipeline.

(II) Dataset description

The *'winequality-red*' dataset comprises data from 1,599 different red wines, characterized by 11 physicochemical features: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, and alcohol. The dataset targets the wine quality on a scale of 0 to 10. Notably, there are no missing values in the dataset, which simplifies preprocessing steps regarding data completeness.

(III) Experimental setup

☐ ***Programming environment and libraries:***

The analysis was conducted using Python, leveraging libraries such as Pandas for data manipulation, Scikit-learn for building ML models and preprocessing, and skopt for advanced hyperparameter optimization.

☐ ***Data preparation and splitting***

Data was randomly divided into training (80%) and testing (20%) subsets, using a 'random_state' of 42. Numerical features were subjected to optional logarithmic transformation and standard scaling (to be decided by pipeline optimizer). These steps were encapsulated within a 'ColumnTransformer**'**, ensuring that different transformations apply to appropriate feature types. The ML pipeline optimization task tunes the preprocessing options (e.g., activation of transformations).

☐ ***ML pipeline configuration***

The pipeline for the ML integrates:

- **Preprocessing**: As described above, with components conditionally applied based on optimization results.
- **Feature selection**: **SelectKBest** selects the top 10 features.
- **Classifier**: A placeholder filled by one of the classifiers (Random Forest, Gradient Boosting, KNN) depending on the experiment phase.

☐ ***Hyperparameter optimization strategy***

The optimization of hyperparameters was conducted using "**Random Search"** and **"Bayesian Optimization**" techniques. Both methods were performed with the following parameters:

- **Iterations**: 30 random combinations per classifier
- **Cross-validation (CV)**: 5-fold, to evaluate generalizability

- **Random state**: 42, to ensure consistent random sampling

These methods explore a broad parameter space efficiently and investigate multiple configurations to identify the most effective ones. The hyperparameters of the classifier's included:

- **Random Forest**: Number of trees, tree depth, minimum samples per leaf
- **Gradient Boosting**: Learning rate, number of estimators, tree depth, minimum samples per leaf
- **KNN**: Number of neighbors, weighting method, neighbor-search algorithm

(IV)Results and observations

The default configurations and the post-optimization results using both Random and Bayesian search methods for each classifier are summarized in Table 1. The illustration of the accuracy values for the test data are also presented in Figure 1.

The results demonstrate that the implementation of pipeline optimization and hyperparameter tuning has led to significant enhancements in model accuracies across all classifiers. Notably, there has been a noteworthy increase in both training and test data accuracies, ranging from 3% to 10% depending on the classifier and optimization technique employed.

For the Random Forest and Gradient Boosting models, the test accuracies were increased by approximately 2-3%. For these two classifiers, Random Search optimization demonstrated a superior performance over Bayesian optimization, which might be attributed to the fact that Random Search explores the hyperparameter space more extensively and is better suited for algorithms with complex and high-dimensional parameter spaces, such as Random Forest and Gradient Boosting. In contrast, Bayesian optimization tends to perform better with algorithms that have smoother, more continuous parameter spaces.

For the KNN classifier, the model initially exhibited poor performance but experienced a remarkable improvement in accuracy (up to 10% on the test data) following the application of Bayesian optimization. This enhancement highlighted the effectiveness of optimization in fine-tuning hyperparameters for ML algorithms.

Table 1. The training and test accuracies before and after optimization of the Random Forest, Gradient Boosting, and KNN classifiers.

Classifier	Default		Random Search		Bayesian Search	
	Test	Training	Test	Training	Test	Training
Random Forest	63.75%	68.73%	65.94%	70.12%	65.62%	70.48%
Gradient Boosting	61.56%	65.13%	64.38%	66.89%	62.81%	67.25%
KNN	56.87%	57.61%	65.31%	66.47%	66.87%	67.82%

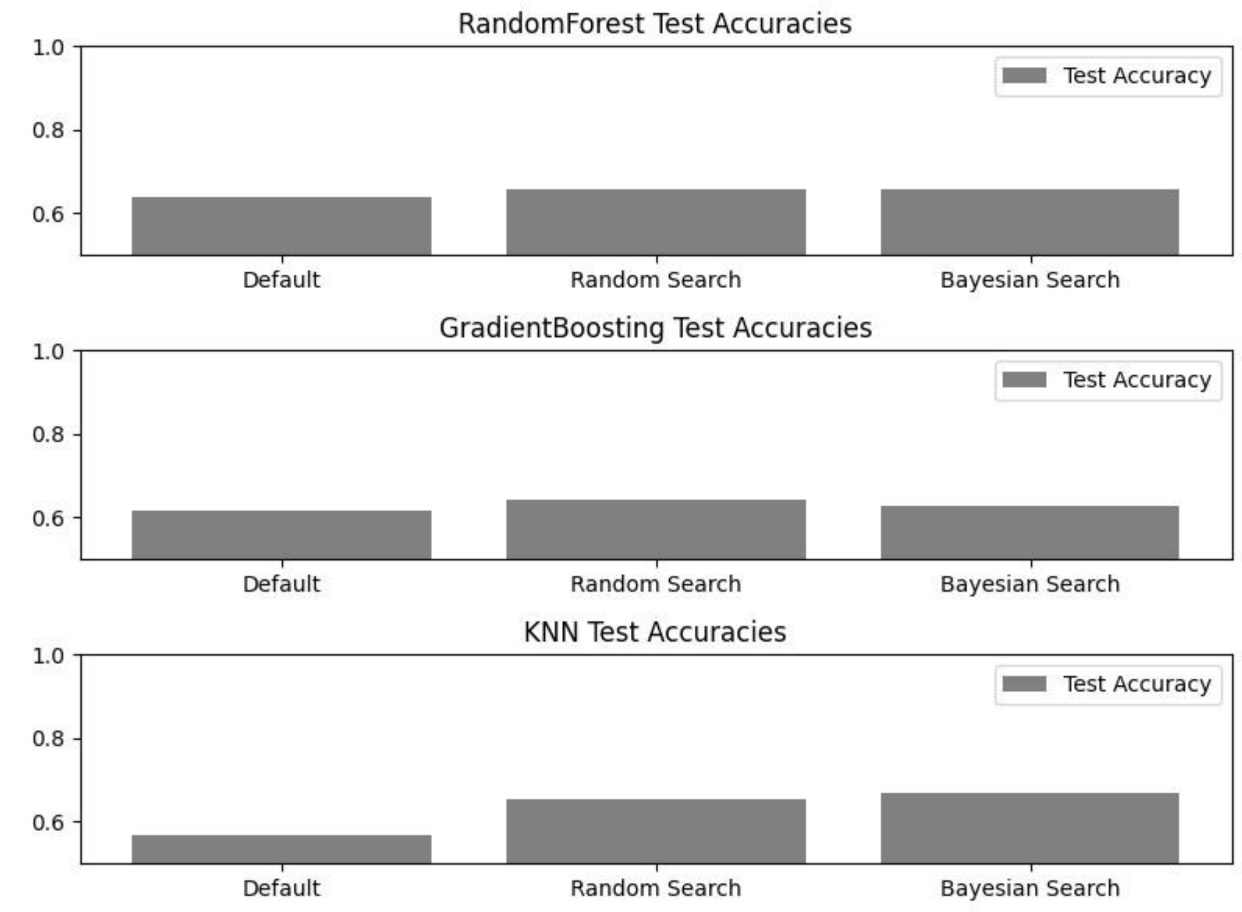

Figure 1: Test accuracies of Random Forest, Gradient Boosting, and KNN classifiers before and after pipeline optimization

(V) Conclusion

This comprehensive experiment demonstrates the significant impact of systematic optimization across the entire ML pipeline. The results also showed that the appropriate hyperparameter optimization technique may vary depending on the nature of the ML model (e.g., Random Search for Random Forest model and Bayesian optimization for the KNN model). By tuning both preprocessing and model parameters together, rather than in isolation, the study provides deeper insights into the interactive effects between the elements of ML pipeline and enhances our understanding of the performance of the predictive models.

(VI) References

1) Konstantinos Filippou. (January 2024). Structure Learning and Hyperparameter Optimization Using an Automated Machine Learning (AutoML) Pipeline.
2) A Comprehensive Guide on Hyperparameter Tuning and its Techniques (https: analyticsvidhya.com)
3) ML Pipeline Architecture Design Patterns(https://neptune.ai/blog/ml-pipeline-architecture-design-patterns)