ML Algorithm Selection
Ali Torabi

1. Introduction

Machine learning is changing our lives swiftly. While a simple task like image recognition seems so trivial for humans, there is a lot of work to do in the machine learning pipeline, such as data cleaning, feature engineering, finding which models best fit for the specific problem, and hyperparameters tuning, among many other tasks. A lot of these tasks are still not automated and need an expert to do some of these trials and errors. Automated Machine Learning (AutoML) provides a process to automatically discover the best machine learning model for a given task according to the dataset with very little expert need. This experiment uses SVM as a default algorithm. Then it will compares the accuracy with/without hyperparameter optimization on GridSearch and BaysianOptimization methods.

2. Dataset Description

The dataset chosen for this experiment is the Wine Quality dataset [1]. It is related to white wine samples from the north of Portugal. It comprises 1599 instances and 11 different features. The label is the quality of the wine that makes the data suitable for Classification and Regression tasks. Each of these algorithms would be to detect the quality of wine ranges from poor to excellent. As mentioned in the description of the dataset itself, it has no missing value. The features are: fixed_acidity, volatile_acidity, citric_acid, residual_sugar, chlorides, free_sulfur_dioxide, total_sulfur_dioxide, density, pH, and sulphates. The output variable is quality which is scored between 0 and 10. We can use the Pandas describe method to show some of the main properties of the dataset.

	fixed acidity	volatile acidity	citric acid	residual sugar	pH	sulphates	alcohol	quality
count	4898	4898	4898	4898	4898	4898	4898	4898
mean	6.854788	0.278241	0.334192	6.391415	3.188267	0.489847	10.51427	5.877909
std	0.843868	0.100795	0.12102	5.072058	0.151001	0.114126	1.230621	0.885639
min	3.8	0.08	0	0.6	2.72	0.22	8	3
25%	6.3	0.21	0.27	1.7	3.09	0.41	9.5	5
50%	6.8	0.26	0.32	5.2	3.18	0.47	10.4	6
75%	7.3	0.32	0.39	9.9	3.28	0.55	11.4	6
max	14.2	1.1	1.66	65.8	3.82	1.08	14.2	9

3. Experimental Setup

In this experiment I use SKlearn library for Hyperparameter optimization. The dataset also split into Train, Validation,z and Test sets. First, I choose SVM as the algorithm to train on dataset. Then, using GridSearch to loop over some of the hyperparameters to find which combination of hyperparameters has the best result in terms of accuracy. Then, the Bayesian Optimization method is used for hyperparameters tuning in SVM and at the end the results has been compared in terms of accuracy.

The most critical hyperparameters for SVM are kernel, C, and gamma. kernel function transforms the training dataset into higher dimensions to make it linearly separable. The default kernel function for the python implementation of the support vector classifier is the Radial Basis Function, which is usually referred to as rbf. C is the l2 regularization parameter. The value of C is inversely proportional to the strength of the regularization. gamma is the kernel coefficient for rbf, poly, and sigmoid. It can be seen as the inverse of the support vector influence radius. The gamma parameter highly impacts the model performance. Gamma can take the value of scale, auto, or a float value. The default value for the python sklearn implementation is scale since version 0.22 [2].

The classification report for SVM without using any hyperparameter optimization method is depicted in table below. As the table shown, the accuracy is 0.46.

	precision	recall	f1-score	support
3	0.00	0.00	0.00	6
4	0.00	0.00	0.00	71
5	0.46	0.03	0.05	568
6	0.46	0.98	0.63	905
7	0.00	0.00	0.00	344
8	0.00	0.00	0.00	66
accuracy			0.46	1960
macro avg	0.15	0.17	0.11	1960
weighted avg	0.35	0.46	0.31	1960

The set of hyperparameters used for optimization in GridSearch is as follows:

No.	Hyperparameter	Designated Values
1	C	0.1, 1, 10, 100, 1000
2	Gamma	1, 0.1, 0.01, 0.001, 0.0001
3	Kernel	Rbf, poly

After searching through all of these hyperparameters values the best result is as the table below:

	precision	recall	f1-score	support
3	0.00	0.00	0.00	6
4	1.00	0.06	0.11	71
5	0.94	0.24	0.38	568
6	0.53	0.99	0.69	905
7	0.99	0.28	0.44	344
8	1.00	0.27	0.43	66
accuracy			0.59	1960
macro avg	0.74	0.31	0.34	1960
weighted avg	0.76	0.59	0.52	1960

Where the accuracy has been increased to 0.59 with hyperparameters as C=1, gamma=1, and kernel=rbf. As it shown, the accuracy is better when we use hyperparameter optimization.

In the next step, SVM is going through hyperparameter tuning using Bayesian Optimization. At then end, the best hyperparameters for the models is C=2, gamma=1 and kernel=rbf with accuracy 62%.

Model	Accuracy
SVM Default	46%
SVM with GridSearch	59%
SVM with Bayesian Optimization	62%

Obviously, the HPO with Bayesian has the best result.

References:

1 - https://medium.com/grabngoinfo/support-vector-machine-svm-hyperparameter-tuning-in-python-a65586289bcb
2 - https://www.geeksforgeeks.org/svm-hyperparameter-tuning-using-gridsearchcv-ml/
3 - https://medium.com/grabngoinfo/support-vector-machine-svm-hyperparameter-tuning-in-python-a65586289bcb#:~:text=You%20can%20check%20out%20the,to%20make%20it%20linearly%20separable.