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 a sample dataset to see how it works with different machine learning models. The aim is to find the best model for this particular dataset.

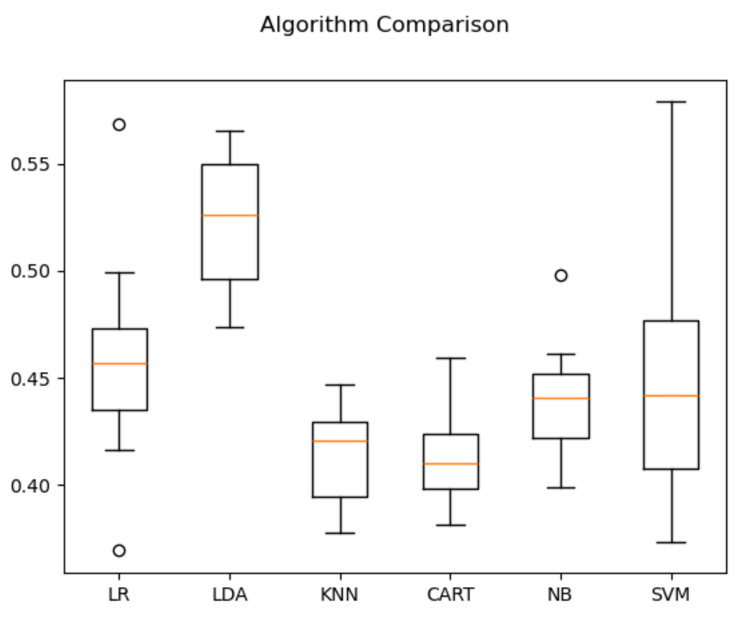
2. Dataset Description

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

For this experiment, the Python programming language (version 3.10) has been chosen. The SKlearn library has been used for cross-validation. The algorithms used for training are Linear Regression, Random Forest, Logistic Regression, Decision Tree, KNeighborClassifier, GaussianNB, and SVM. After going through the training phase, true accuracy has been shown in the test phase. For the sake of generality and to make the comparison fair, the same default parameters have been selected for all of the algorithms. For cross-validation, the k-fold with n_splits = 10 has been chosen in order to split the dataset into different categories as train, validation, and test. So, for every model, we have ten different scores, and the mean and standard deviation have been computed as shown in the boxplot below.



For each of those models, the maximum score as accuracy is shown in the table below:

Algorithm	Accuracy (In Percentage)
Logisitc Regression	%56
Linear Discriminant Analysis	%56
KNeighborClassifier	%44
Decision Tree	%45
GaussianNB	%49
SVM	%57

In this experiment, SVM has the highest accuracy. The last experiment is more realistic because I used K-Fold cross-validation for all of the different algorithms, and the accuracy test results are more plausible. With using the 10-fold cross-validation score, the best test accuracy result is SVM, and the worst one is KNeighbor Classifier. The reason why the experiment with 10-Fold cross-validation is more realistic is that it seems like throwing a dice more than once (10 times) and then computer accuracy by averaging them instead of only computing one accuracy. So, K-Fold cross-validation isn't for increasing the accuracy, it's a method to provide a more accurate measure.