ML Pipeline Optimization

COSC 5010 – 03

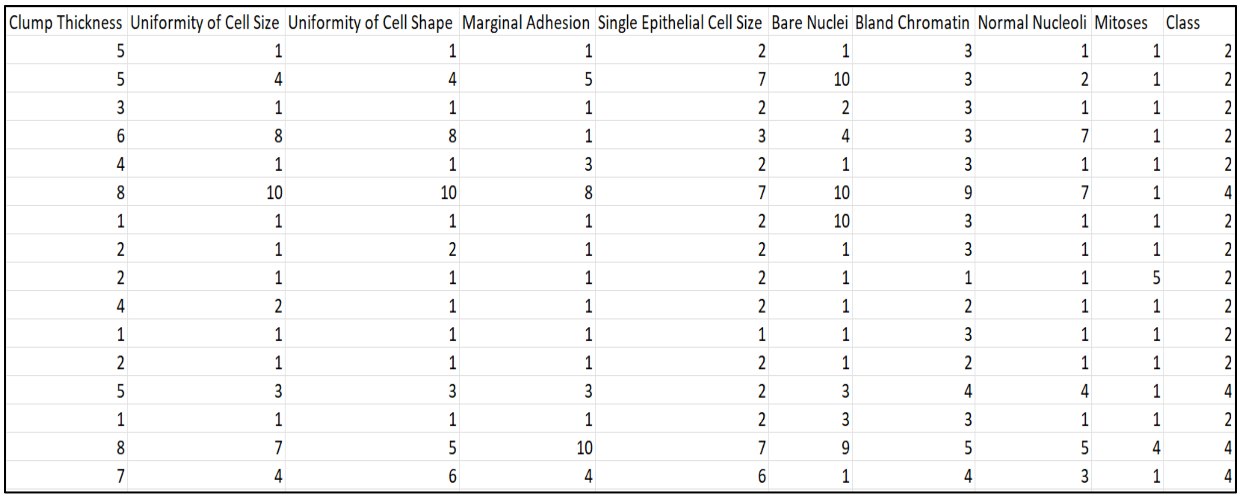
Isaac Baah

1.0 Introduction

Building a pipeline in machine learning to make some preprocessing stages of the model building is necessary to make the work of engineers smooth and streamlined. In the project, we look at building a machine learning pipeline for hyperparameter optimization. The machine learning task is a classification problem where we are trying to build a ML pipeline to predict whether based on the characteristics of a plant will grow "big or tall" or otherwise.

2.0 Data

The data used is from a machine learning tutorial I did on Udemy, the data consists of nine independent variables. All the variables are numeric. The dependent variable is the "Class". The data is trying to predict the whether the plant will grow "big or tall" or otherwise considering the characteristics of the plant. Class "2" means the plant will grow "big or tall" and Class "4" means "Otherwise".



Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	Class
5	1	1	1	2	1	3	1	1	2
5	4	4	5	7	10	3	2	1	2
3	1	1	1	2	2	3	1	1	2
6	8	8	1	3	4	3	7	1	2
4	1	1	3	2	1	3	1	1	2
8	10	10	8	7	10	9	7	1	4
1	1	1	1	2	10	3	1	1	2
2	1	2	1	2	1	3	1	1	2
2	1	1	1	2	1	1	1	5	2
4	2	1	1	2	1	2	1	1	2
1	1	1	1	1	1	3	1	1	2
2	1	1	1	2	1	2	1	1	2
5	3	3	3	2	3	4	4	1	4
1	1	1	1	2	3	3	1	1	2
8	7	5	10	7	9	5	5	4	4
7	4	6	4	6	1	4	3	1	4

Figure 1. A snapshot of the data used in the project.

3.0 Experimental Setup

To predict based on the characteristics of the plant whether a plant will grow very well "big or tall" or grow otherwise, several models were developed. In this project, the following models were developed, Logistic Regression, Support Vector Machines, Decision Trees, and Random Forest Classifier. This project used the scikit-learn library available in python to conduct this project. Using the decision tree as an example for fitting the data. The first thing was to split the data into training and testing datasets. The data was split into 80% training data and 25% testing data. The first step was to create a dictionary of all the models and their hyperparameters. For this project, I only selected a few ranges for the hyperparameter tuning since I decided to use "GridSearchCV" this time around. My laptop takes a lot of time to run when I use "GridSearchCV" and a wide range of hyperparameters.

The next step was to fit the model on the training data and print out the best hyperparameters for optimal performance. GridSearchCV hyperparameter optimization method with a cross-validation score of 5 was used in this project.

4.0 Results

In this section, I present the results of the ML pipeline optimization project. Among all the hyperparameters included in the model, the best hyperparameters to ensure that we have an optimal model performance is presented in Table 1 below. The rest of the hyperparameters were kept as default.

Table 1. Selected Hyperparameters for the final models

ML Models	**Best Score (%)**	**Hyperparameters**
Support Vector Machines	97.26	C = 1, Kernel = rbf
Logistic Regression	96.68	C = 1
Decision Trees	96.68	criterion = 'log_loss'
Random Forest	94.92	n_estimators = 10

5.0 References

1. Hands-On Machine Learning with Python and R. https://www.udemy.com/course/machinelearning/. [Accessed, October 2023].

2. Stack overflow. https://stackoverflow.com/questions/72854063/making-pipeline-for-machine-learning-models. [Accessed, December 13]