[DESAFÍO #1] Machine Learning & MLOps by: Daniel Fernando Barrera Armendaris

Index

Introduction

For this project a classification model for varieties of wines is requested. We receive a dataset from a URL, this is downloaded and start a process of cleaning, standardization, reduction, training and evaluating of ML models. In this case I have tried with three different models. SVN, KNN and naives bayes. These models are pre-selected because of the size of the registers and to try with different approaches.

Structure of notebooks

The project has the following structure:

config:
In this notebook is downloaded the dataset that is going to be used

1_eda_nb:
In this notebook is done a simple eda to find some patterns and the status of the data.

2_data_processing_nb:
In this notebook is done the processing of data to make the data more suitable for the models that we train.

3.1_svm_mlflow:
In this notebook is done the first experiment with the algorithm SVM (Support vector machine)

3.2_knn_mlflow:
In this notebook is done the first experiment with the algorithm KNN.

3.3_naive_bayes_mlflow:
In this notebook is done the first experiment with the algorithm naive bayes.

4 _get predictions:
In this notebook is done the predictions after the training of the previous models

Dimensionality reduction

I will try with this approach because of the heatmap in the EDA. I can see that there are some variables that can be redundant because of the highly correlations. In addition, it works as well to standardize the values and improve the performance of the models. We try techniques to see which is the best number of components for the PCA algorithm.

Selection of models

To select the best model i check the following metrics:

accuracy:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

f1_score:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

precision:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

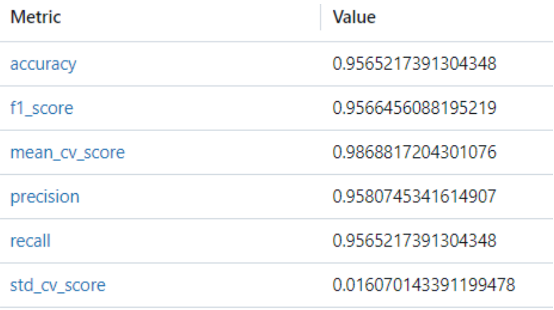
recall:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

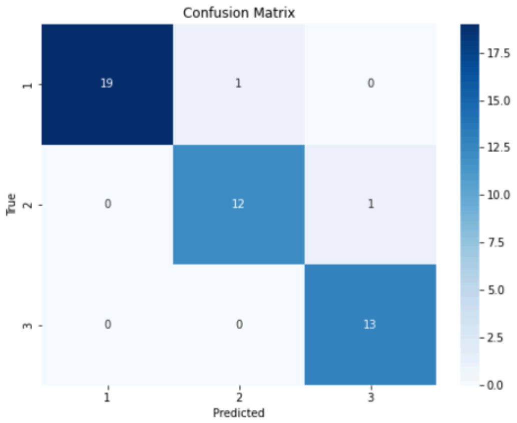
standard_deviation in cross validation:
This is important to see how the performance of the model takes different training and validation datasets. I use that to check the performance of it.

mean in cross validation:
It is useful for getting a robust estimate of the model's performance compared to using a single data partition.

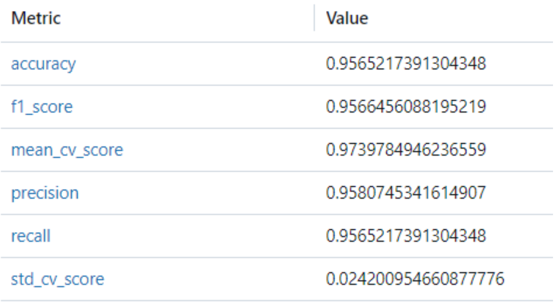
Naive bayes



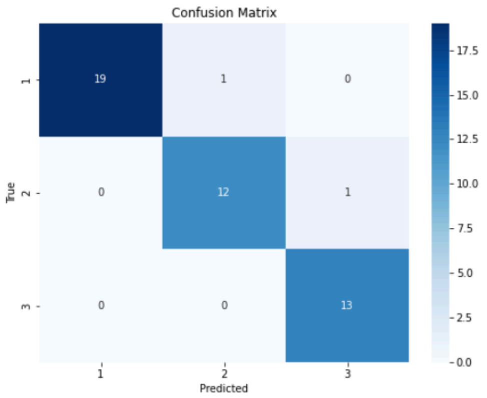
Metric	Value
accuracy	0.9565217391304348
f1_score	0.9566456088195219
mean_cv_score	0.9868817204301076
precision	0.9580745341614907
recall	0.9565217391304348
std_cv_score	0.016070143391199478



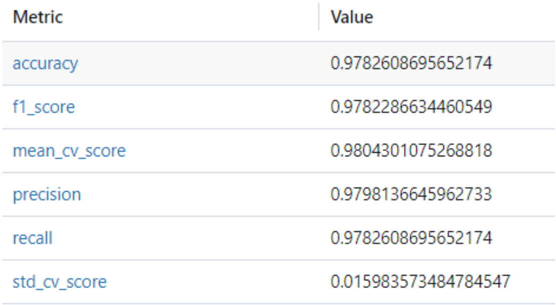
KNN



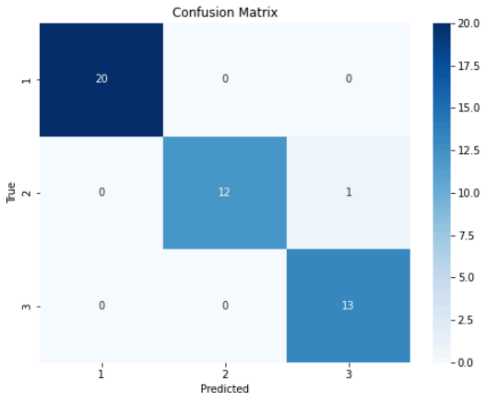
Metric	Value
accuracy	0.9565217391304348
f1_score	0.9566456088195219
mean_cv_score	0.9739784946236559
precision	0.9580745341614907
recall	0.9565217391304348
std_cv_score	0.024200954660877776



SVM



Metric	Value
accuracy	0.9782608695652174
f1_score	0.9782286634460549
mean_cv_score	0.9804301075268818
precision	0.9798136645962733
recall	0.9782608695652174
std_cv_score	0.015983573484784547



Having the previous metrics into consideration, the model with the highest values is the SVM. In addition, the number of true predictions is more effective in its confusion matrix.

Technical debt

although in this project were considered several aspects like cleaning data, dimensionality reduction, eliminating of outliers, the training of the models. The best model was selected in an empiric form. There are more aspects to see for example. Is this a case of overfitting/underfitting? the hyperparameters the best that we can get? Does the algorithm work for bigger registers?