Serechenko Alexandra

Big homework

Type: lazy FCA

13.12.22

Description of the dataset

The dataset describes 22 physical features (such as stalk-root, gill-shape, cap-color, habitat, etc) of 8124 mushrooms. The set of mushrooms consists of two classes: either poisonous or edible (target variable). Detailed description of the features is available via the dataset page (Rijn, 2014)[1]. All 22 features are categorical variables.

For this homework, it was required to use a small dataset - with several hundred lines. Therefore, I randomly selected 1000 lines from my dataset. Next, I will test all the algorithms on this dataset. Then I moved on to binarization of variables. First, I changed the encoding of the target variable: if the mushroom is edible, then the value became true, if the mushroom is poisonous, then false. Also, using the function from the file, I binarized all the features. Initially there were 22 columns with features, after binarization of columns (with features) increased to 112.

Algorithm 1

First we look at the positive context. We take the intersection of these elements and the test one. For each element of the negative context, we check whether the intersection of it and the test object is the same as for the test object and the positive context. If is, then we have found a counterexample for the positive class. The same is true for negative ones, we count the number of counterexamples. If there are fewer counter-examples for the positive than for the negative, then we predict the positive, otherwise — the negative.

Prediction quality measure

In addition to the most popular prediction quality metrics: accuracy and f1_score, I decided to consider more: precision and recall. The precision for a class is the number of true positives (i.e. the number of items correctly labelled as belonging to the positive class) divided by the total number of elements labelled as belonging to the positive class (i.e. the sum of true positives and false positives, which are items incorrectly labelled as belonging to the class). Recall in this context is defined as the number of true positives divided by the total number of elements that actually belong to the positive class.

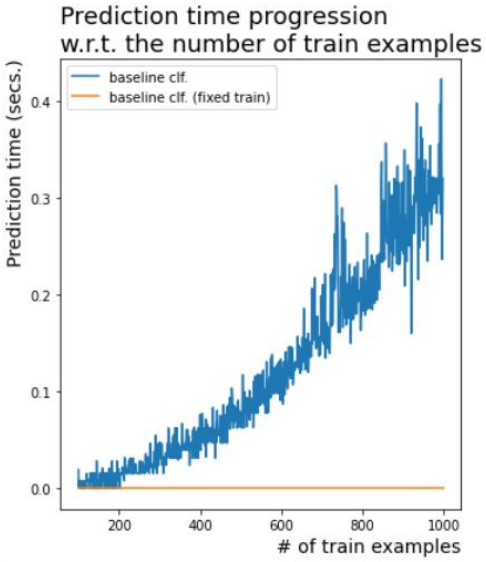
[1] Jan van Rijn (2014). mushroom. https://www.openml.org/search?type=data&sort=runs&status=active&id=24

Taking into account additional metrics will allow us to evaluate the quality of the algorithm prediction in more detail.

Quality metrics for algorithm 1:

accuracy	94.71
f1_score	95.35
recall_score	98.58
precision_score	92.33

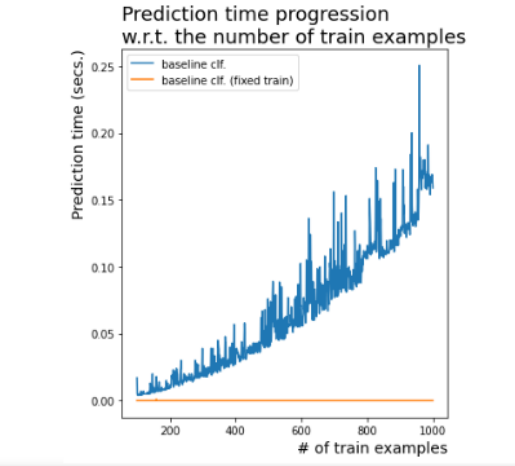
Thus, in terms of prediction quality, the original algorithm is quite good. All quality metrics are close to 1. However, in terms of execution time, the algorithm is unsuccessful. It works fast enough on a small data set, but when adding a large amount of data, the speed of completion is noticeably reduced.



Algorithm 2

I also decided to rewrite the original algorithm not using python loops, but using numpy

accuracy	95.18
f1_score	95.32
recall_score	97.36
precision_score	93.37



The new algorithm is slightly better both in terms of quality metrics and in terms of its execution time.

Algorithm 3

I also decided to consider another algorithm. We have divided our training sample into positive and negative examples. Now, for each observation, it is necessary to calculate how many characteristics in it that are characteristic of positive examples, and how many characteristics that are characteristic of negative examples. If there are more positive ones, then we will attribute this observation to positive examples, otherwise to negative ones.

accuracy	42,1
f1_score	0
recall_score	0
precision_score	0

Despite the fact that the speed of this algorithm is better, the quality metrics of this algorithm are terrible.

Algorithm 4 and 5

I also decided to build classical machine learning models: DecisionTree and RandomForest. The accuracy for DecisionTree algorithm is 99,12 , and for RandomForest is 100.