Assignment 2 - Training and evaluating a stacked classifier

March 20, 2024

Simone Vaccari
915222

1 Objective

The objective of this experiment is to analyze the performances of a stacked classifier, a particular case of ensemble learner trained on the predictions of multiple classifiers (called 'level-one classifiers'), and implement a case of "incorrect" training procedure.

2 Procedure

To perform our analysis we worked with a training set (shown in Figure 1) consisting of samples belonging to two different classes.

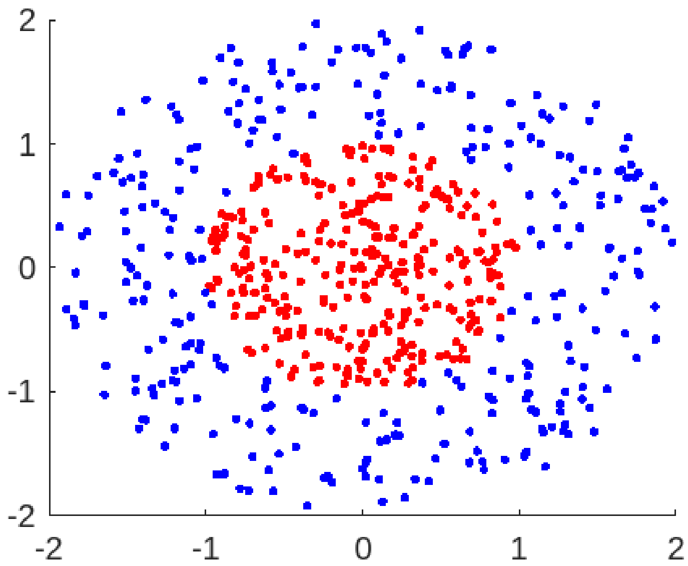

Figure 1: Training set used for the analysis

The first step was to split the training data into two folds of equal size, since when classifiers are stacked it is important that the predictions of the level-one classifiers come from a subset of the data that is not used to train the level-one classifiers themselves.

As a consequence, our chosen models were trained on one of the folds and used to make predictions on the other one.

The learners that were used as level-one classifiers are the following:

- SVM, with gaussian kernel and scale factor 5;
- SVM, with polynomial kernel and scale factor 10;
- Decision Tree, with Gini Index as splitting criterion and 20 as the maximum number of splits;

- Naive Bayes classifier;
- Ensemble of Decision Trees;

These 5 models were used to produce predictions and classification scores. The latter were then used to train the stacked classifier (also called **meta-classifier**), defined as an ensemble learner using the **"Bagging"** method.

Once all the models were trained, we used the level-one classifiers to make predictions on the testing data, computing their accuracy, and with the yielded scores we tested the meta-classifier, obtaining its value of accuracy.

Then, the same procedure was carried out but this time the stacked classifier was trained and evaluated **directly on the predictions** made by the base learners, rather than on the classification scores, leading to a decrease in the performance.

As a last experiment, we tried to adopt the unwise approach of not splitting the training data at the beginning, thus training the level-one learners and the meta-classifiers on the same data, and repeated the whole process, with the purpose of comparing the results.

3 Results

The accuracies of the level-one classifiers obtained in the first case, when the training data was split, using part of it to train the base learners and the rest to make predictions, are shown in the following table:

Gaussian SVM	**Polynomial SVM**	**Tree**	**Naive Bayes**	**DT Ensemble**
0.8683	0.6250	0.9483	0.9783	0.9533

For the stacked classifier we obtained these accuracies:

	Stacked Classifier
Trained on scores	0.9900
Trained on predictions	0.9700

In the second case, when the data splitting was not performed, the yielded accuracies were:

Gaussian SVM	**Poly SVM**	**Tree**	**Naive Bayes**	**Ensemble of DT**	**SC-scores**	**SC-predictions**
0.9000	0.6333	0.9667	0.9917	0.9683	0.9700	0.9683

where the last two columns indicate the stacked classifier trained on scores and predictions respectively.

4 Conclusions

The first thing we notice is that the performance of the meta-classifier gets worse when trained directly on the level-one predictions, with respect to when it's trained on the classification scores (from 0.99 the accuracy drops to 0.97). This is due to the fact that predictions alone carry less information with respect to the scores, as they only yield the predicted labels.

We can see that we have a drop in the accuracy also in the second case, when we adopt the "incorrect" approach: once again accuracy goes from 0.99 to 0.97, when the meta-classifier is trained on the scores, and, as expected, it drops slightly more (to 0.9683) when the meta-classifier is trained on the predictions. The fact that the accuracy of each level-one classifier increases in the second case is due to the fact that they are trained on the entire training set, rather than just half of it.