Machine Learning and Data Mining project: Leaf Classification

Michele Alessi[1], Samuele D'Avenia[2], and Elena Rivaroli[3]

[1, 2, 3] problem statement, solution design, solution development, data gathering, writing

Course of AA 2022-2023 - Data Science and Scientific Computing

1 Problem statement

Leaf classification is the problem of assigning the correct tree species to an unknown observed leaf. Traditionally, this was the job of experts called taxonomists. Through the usage of modern automatic recognition systems, it is possible to develop an efficient tool to help with this classification process. If some examples, i.e. leaves with the correct species assigned, are provided, this can easily be seen as a supervised machine learning problem.
The purpose of this project is to propose a leaf classifier after comparing different techniques. This is done using a dataset of leaves containing some of their attributes, which are further described below.

2 Assessment and performance indexes

For choosing the effectiveness index, the following have to be taken into account: this is a multi-class classification problem and the dataset is not perfectly balanced.
The chosen assessment measure is *balanced accuracy* bA, which is defined as follows [2]:

$$bA = \frac{\sum_{i=1}^{n} \frac{|TC_i|}{|C_i|}}{n}$$

where $C_i,\ i = 1, \dots, n$ are the n classes in the dataset D and TC_i is the set of observations correctly predicted into $C_i,\ i = 1, \dots, n$. Since each class recall is computed disregarding other classes, it's clear that bA gives the same weight to each class; this means that smaller classes have the same influence on the final effectiveness as the biggest one; in particular bA is insensitive to imbalanced class distribution. It is important to report the existence of other effectiveness indexes suitable for multi-class classification problem with imbalanced class,

which have been analyzed as well [3]. The final choice fell onto the balanced accuracy for explainability , interpretability and efficiency reasons. It is fast to compute and on equal terms it is the simplest index to interpret.

3 Proposed solution

The proposed solution uses a nested cross-validation approach, which is recommended when dealing with small datasets [4]. The outer cross-validation is used to compare the different models, using an appropriate effectiveness measure to choose the best one (i.e. performance index described in Section 2). The internal nested cross-validation is used to tune the hyper-parameters using the same effectiveness measure (more details on the choice of hyper-parameters are provided in section 4.2) through a grid-search approach.
Since the dataset is not large and slightly imbalanced, both cross-validations are performed using stratification, to obtain better estimates [4]. The number of folds in the inner and outer loop are chosen to ensure the presence of at least one observation of each class in each inner fold.
The description provided below reports the procedure to tune the hyper-parameters of a learning technique $(f'_{\text{learn}}, f'_{\text{predict}})$ and to obtain the effectiveness measure v_{effect} to compare the various effectiveness techniques and select the most appropriate one.

1. Split the whole dataset *data* into k_{outer} folds using stratified splitting.
2. For $i = 1, 2, ..., k_{\text{outer}}$:
   (a) Consider all folds except fold i. Denote the subset of the data containing all folds but i as $data_{(-i)}$.
   (b) Select a finite set of values for each hyper-parameter.
   (c) To perform grid search, for each possible combination of hyper-parameters p repeat:
      i. For $j = 1, 2, ..., k_{\text{inner}}$:
         A. Consider all folds except fold j. Denote the subset of $data_{(-i)}$ containing all folds but j as $data_{(-i,-j)}$.
         B. Learn a model on $data_{(-i,-j)}$ using the hyper-parameters p of the current iteration.
         C. Obtain the value of effectiveness index on fold j, denote it as v_{j}.
      ii. Compute the average effectiveness with this set of hyper-parameters p denoted as $v_p = \text{mean}_j(v_{\text{j}})$.
   (d) Choose as best hyper-parameters the combination which produced the largest v_p.
   (e) Train the model with the best hyper-parameters on $data_{(-i)}$.

(f) Compute $v_{\text{eff, i}}$, the effectiveness measure obtained by applying the learnt prediction function on fold i.

3. Compute v_{eff} as mean of the effectiveness measures obtained $v_{\text{eff, i}}$.

The learning technique with the highest v_{effect} is chosen as the best one.
The final step is to choose the best hyper-parameters for the selected technique using a simple cross-validation on the whole dataset. The v_{effect} obtained in the nested cross-validation as final effectiveness measure.

4 Experimental evaluation

4.1 Data

The leaf dataset used for this task comprises 340 leaf observations from 30 different plant species. For each leaf the *Class*, the *Specimen Number* and 14 different features are available. Variable *Class* contains the plant species of each leaf and is used as the label of the supervised learning technique. Further description of the variables can be found at [5].

4.2 Procedure

The variable *Specimen Number* is removed from the dataset as it is only used as an identifier for each leaf within each class. The dataset was prepared with the task of leaf classification in mind, as such no further feature engineering is deemed necessary.
The techniques tried are decision tree, random forest (RF), k-nearest neighbours (kNN) and support vector machines (SVM) with Gaussian and linear kernel as they are some of the most widely used. For the last three techniques (the two SVM and kNN), standardization is performed to ensure all dependent variables are on the same scale. It is performed in each inner fold of the procedure to avoid the algorithm having knowledge of unseen data (i.e. left out fold data).
Moreover, in order to allow SVM to work with the multi-class problem at hand, a one-vs-all approach is taken for efficiency reasons.
For random forest, the default number of variables to consider at each split ($\sqrt{p}$ where p is the number of variables) is deemed satisfactory. As such no tuning is performed for that hyper-parameter. For similar reasons, Gini criterion is used as node impurity measure both for decision tree and random forest [6].
Table 1 below reports what hyper-parameters are included in the grid search procedure (along with a brief description) for each technique. It also contains the results of the nested cross-validation procedure (outlined in Section 3) in the last two columns. These two represent the mean effectiveness score v_{effect} and the standard deviation (s.d.).

Learning technique	Hyper-parameters	v_{effect}	s.d.
Decision tree	$n_{\min}$: minimum number of samples required to be at a leaf node	0.63	0.062
Random forest	n_{tree}: number of trees in the forest	0.79	0.035
SVM Gaussian kernel	γ: kernel coefficient c: regularization parameter	0.79	0.059
SVM linear kernel	c: regularization parameter	0.80	0.059
KNN	k: number of neighbours d: type of distance	0.69	0.010

Table 1: Results of the nested cross-validation.

It appears that the RF and SVM with Gaussian and linear kernel algorithms perform slightly better. In fact they have a larger mean balanced accuracy. However, RF seems to have a smaller standard deviation.
It also requires tuning of only one hyper-parameter which is easily interpretable. As such it is chosen as the best classifier for this problem.
Another advantage of using Random Forests is that since it is an ensemble technique, out-of-bag (OOB) trees can be used to provide another estimate of effectiveness measure [1].
The final RF is fitted on the whole data and hyper-parameter tuning is performed using a grid-search approach. From this fitting procedure the OOB estimate for the balanced accuracy and the recall for each class is obtained.

4.3 Results and discussion

With the final grid-search procedure, the selected number of trees is $n_{\text{tree}} = 500$. The model weighted accuracy v_{effect}, obtained through the nested cross-validation, is 0.79.
The same index is also computed using OOB estimation. In this case a weighted accuracy of 0.76 is achieved. This is quite similar to the one obtained previously. Through OOB error estimation, it is also possible to compute an estimate of which species our classifier correctly identifies and which it does not.
It appears that the classes which are misclassified the most are class 4 (i.e. *Alnus*) and class 32 (i.e. *Acca Sellowiana*). To ensure this was not dependent on the random component of the model, a set of different seeds are chosen and the results were always the same.

However, if there were more data available, an estimate using a test set would have been more appropriate.

References

[1] Leo Breiman. *Out-of-Bag estimation*. Technical report, Dept. Statistics, Univ. California, Berkeley, CA., 1997.

[2] Margherita Grandini, Enrico Bagli, and Giorgio Visani. *Metrics for multi-class classification: an overview.* arXiv:2008.05756, 2020.

[3] Akhilesh Gupta, Nesime Tatbul, Ryan Marcus, Shengtian Zhou, Insup Lee, and Justin Gottschlich. *Class-weighted evaluation metrics for imbalanced data classification.* arXiv:2010.05995, 2020.

[4] Sebastian Raschka. *Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning.* arXiv:1811.12808, 2018.

[5] Pedro Filipe Silva. *Development of a System for Automatic Plant Species Recognition.* Master's thesis, Faculdade de Ciencias da Universidade do Porto, 2013.

[6] Fabian Spaeh and Sven Kosub. *Global Evaluation for Decision Tree Learning.* arXiv.2208.04828, 2022.