

Semi-supervised classification trees

Jurica Levatić^{1,2}  · Michelangelo Ceci³ ·
Dragi Kocev^{1,2} · Sašo Džeroski^{1,2}

Received: 11 July 2016 / Revised: 28 December 2016 / Accepted: 15 March 2017
© Springer Science+Business Media New York 2017

Abstract In many real-life problems, obtaining labelled data can be a very expensive and laborious task, while unlabeled data can be abundant. The availability of labeled data can seriously limit the performance of supervised learning methods. Here, we propose a semi-supervised classification tree induction algorithm that can exploit both the labelled and unlabeled data, while preserving all of the appealing characteristics of standard supervised decision trees: being non-parametric, efficient, having good predictive performance and producing readily interpretable models. Moreover, we further improve their predictive performance by using them as base predictive models in random forests. We performed an extensive empirical evaluation on 12 binary and 12 multi-class classification datasets. The results showed that the proposed methods improve the predictive performance of their supervised counterparts. Moreover, we show that, in cases with limited availability of labeled data, the semi-supervised decision trees often yield models that are smaller and easier to interpret than supervised decision trees.

Keywords Semi-supervised learning · Binary classification · Multi-class classification · Decision trees · Random forests

✉ Jurica Levatić
jurica.levatic@ijs.si

Michelangelo Ceci
michelangelo.ceci@uniba.it

Dragi Kocev
dragi.kocev@ijs.si

Sašo Džeroski
saso.dzeroski@ijs.si

¹ Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia

² Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

³ Department of Computer Science, University of Bari Aldo Moro, Bari, Italy

1 Introduction

Traditional supervised algorithms often require a large amount of labeled data to learn models with good predictive performance. However, in many real life problems, labeled data are expensive and hard to get since the labeling procedure may require human experts, expensive devices or time consuming experiments. Real-world classification problems of this type include: phonetic annotation of human speech, protein 3D structure prediction, quantitative structure-activity relationship and spam filtering. In these cases, often only a few labeled examples are available to learn from, while on the other hand, unlabeled data are freely available in vast amounts.

The concept of semi-supervised learning (SSL) emerged in the 1970's as an answer to the problem of labeled data scarcity (Chapelle et al. 2006). Semi-supervised algorithms use unlabeled examples, in addition to labeled ones, in order to learn models with better performance than the models learnt by using only labeled examples. However, to exploit this advantage, it is necessary that the knowledge we gain through unlabeled data has to carry some information about the class labels. If this prerequisite is fulfilled, we can draw on unlabeled data by making certain assumptions about the behavior of labels with respect to the structure of unlabeled data.

Since SSL is considered an established research topic, a number of different semi-supervised methods have been proposed in the literature. On the basis of the assumption(s) they implement, SSL methods can be grouped as follows (Zhu 2008): generative models, low-density separation methods, graph-based methods, and self-training and similar methods (e.g., co-training).

Generative models assume a probabilistic model of the data and use unlabeled, together with the labeled data, to estimate the most probable model parameters (e.g., the Expectation-Maximization algorithm (Nigam et al. 2000)). Differently, low-density separation methods assume that the decision boundary should lie in the region of low density of the data (e.g., semi-supervised support vector machines (Bennett et al. 1999)). Equivalent to the low density separation assumption is the cluster assumption: the points belonging to the same cluster should be of the same class. Graph-based methods use nodes for data representation (labeled and unlabeled) and edges for propagation of the labels through the graph, assuming label smoothness over the graph (e.g., linear neighborhood propagation method (Zhang and Wang 2009)). Self-training (Yarowsky 1995), co-training (Blum and Mitchell 1998) and other similar approaches (Triguero et al. 2015) use their own most reliable predictions in the training process (assuming they are correct), as additional data for learning.

Despite the number of semi-supervised methods already proposed, only few of them exploit the desirable properties of decision trees: they are non-parametric, efficient, readily interpretable, yet exhibit excellent predictive performance in many domains (Breiman et al. 1984; Quinlan 1993; De'ath and Fabricius 2000; Rokach and Maimon 2014). Furthermore, interpretable semi-supervised methods hardly exist. Majority of semi-supervised approaches that incorporate decision trees are based on self-training or similar paradigms. While some researchers reported success with such methods (Tanha et al. 2015), the others show evidence that, in some cases, self-training can seriously deteriorate the performance of decision trees (Guo et al. 2010). The main pitfall of self-training is the reinforcement of mistakes - a mistake once made can reinforce itself in the next iterations, leading to a degradation of predictive performance. The other issue of self-training is the need to repetitively re-train the base model, which considerably increases the computational cost.

In this work, we propose an algorithm for semi-supervised learning of classification trees. In contrast to self-training, in the proposed algorithm, unlabeled examples are exploited

directly in the tree construction process to improve the quality of the splits. This is achieved by relying on the aforementioned cluster assumption. The proposed extension preserves all of the appealing characteristics of decision trees, and, at the same time, enables the use of unlabeled examples. Moreover, the empirical evaluation on several binary and multi-class classification benchmark datasets reveals that the proposed SSL trees yield better predictive performance. Furthermore, we investigate whether the improvement in predictive performance carries over in the ensemble setting, i.e., whether an ensemble of SSL trees achieves better predictive performance than an ensemble of regular supervised trees.

The remainder of this paper is organized as follows. Section 2 describes the proposed methods, while Section 3 specifies the experimental design. The results of the empirical investigations are presented and discussed in Section 4. Section 5 presents the related work. Finally, Section 6 concludes the paper.

2 Methods

2.1 Semi-supervised learning of classification trees

Semi-supervised algorithms exploit unlabeled data by making certain assumptions about the behavior of labels with respect to the structure of unlabeled data (Chapelle et al. 2006). The semi-supervised approach to learn classification trees proposed in this paper is motivated by the popular semi-supervised cluster assumption: *If examples are in the same cluster, then they are likely of the same class*. Actually, a tree can be considered as a hierarchy of clusters, thus our aim is to design decision trees where labeled and unlabeled examples similar to each other are clustered together. If the cluster assumption holds, labels of unlabeled examples should be coherent with the labels associated to labeled examples falling in the same leaves of the tree.

In traditional supervised trees, the quality of splits (i.e., the impurity of splits) is evaluated only on the basis of the target space, i.e., the class labels. For example, by identifying the split that maximizes the information gain. The effect is that the resulting clusters are homogeneous with respect to the class label. Our goal is to learn trees that produce clusters which are homogeneous both in the descriptive and in the target space. To achieve this, we need a different measure of impurity. Specifically, we propose to measure the impurity by considering the similarity among examples on the basis of both the class labels and descriptive attributes. Since the class label is unknown for unlabeled examples (i.e., only descriptive attributes are known), such measure should exploit, during the tree construction, labels associated to examples only if they are available.

To implement the described semi-supervised learning of classification trees, we use the predictive clustering (PC) framework (Blockeel et al. 1998; Koccev et al. 2013). The methods within the PC framework partition the instances into subsets, and can thus be considered as both predictive and clustering methods. These methods include the learning of decision trees and decision rules.

In particular, the PC framework views a decision tree as a hierarchy of clusters, where the top-node corresponds to one cluster containing all the data. This cluster is recursively partitioned into smaller clusters while moving down the tree. The PC framework is implemented in the CLUS system (Koccev et al. 2013), available at <http://sourceforge.net/projects/clus>.

The main difference between the algorithm for learning predictive clustering trees (PCTs) and a standard decision tree learner is that the former considers the impurity function (i.e., the measure of quality of the splits) and the prototype function (that associates

a label to each leaf) as *parameters*, i.e., that can be instantiated for a given learning task. This is an aspect that makes the PCTs suitable for our purpose since, as stated before, the proposed semi-supervised solution requires a novel measure of impurity. So far, PCTs have been instantiated for various machine learning tasks, including multi-target prediction (Struyf and Džeroski 2006), hierarchical multi-label classification (Vens et al. 2008; Levatić et al. 2014), and prediction of time-series (Slavkov et al. 2010).

The semi-supervised approach proposed in this study is based on the standard *top-down induction of decision trees* (TDIDT) algorithm (see Table 1), which takes as input a set of examples E and outputs a tree. The heuristic (h) that is used for selecting the tests (t) to put in internal tree nodes is reduction of impurity caused by partitioning (\mathcal{P} , Table 1, line 3 of the BestTest procedure) the examples according to the tests. In principle, by maximizing the impurity reduction, the cluster homogeneity is maximized and the predictive performance is improved.

In classical supervised PCTs, the impurity for each set of examples E is calculated as the Gini impurity (Table 1, line 4 of the BestTest procedure):

$$\text{Impurity}(E) = \text{Gini}(E, Y). \quad (1)$$

The Gini impurity of a set E for the target variable Y is calculated as follows:

$$\text{Gini}(E, Y) = 1 - \sum_{i=1}^C p_i^2, \quad (2)$$

where C is the number of classes of the target variable Y (e.g., if Y is binary, then $C = 2$), and p_i is the apriori probability of a class c_i (i.e., the proportion of examples in a set E belonging to the class c_i).

The first extension to the classical algorithm for the induction of PCTs concerns the input: the dataset we consider (see Table 1) takes as input the whole set of examples, that is, labeled and unlabeled examples. This means that $E = E_l \cup E_u$, where E_l is the part of the dataset with known labels and E_u is the part with unknown labels. The second extension concerns, as mentioned before, the impurity function which should take into account both the class labels (i.e., the target attribute) and the descriptive attributes in the identification

Table 1 The top-down induction algorithm for decision trees construction

procedure InduceTree

Input: A dataset E

Output: A predictive clustering tree

- 1: $(t^*, h^*, \mathcal{P}^*) = \text{BestTest}(E)$
- 2: **if** $t^* \neq \text{none}$ **then**
- 3: **for each** $E_i \in \mathcal{P}^*$
- 4: $\text{tree}_i = \text{InduceTree}(E_i)$
- 5: **return**
 $\text{node}(t^*, \bigcup_i \{\text{tree}_i\})$
- 6: **else**
- 7: **return** leaf(Prototype(E))

procedure BestTest

Input: A dataset E

Output: the best test (t^*), its heuristic score (h^*) and the partition (\mathcal{P}^*) it induces on the dataset (E)

- 1: $(t^*, h^*, \mathcal{P}^*) = (\text{none}, 0, \emptyset)$
- 2: **for each** possible test t **do**
- 3: $\mathcal{P} =$ partition induced by t on E
- 4: $h = \text{Impurity}(E) - \sum_{E_i \in \mathcal{P}} \frac{|E_i|}{|E|} \text{Impurity}(E_i)$
- 5: **if** $(h > h^*) \wedge \text{Acceptable}(t, \mathcal{P})$ **then**
- 6: $(t^*, h^*, \mathcal{P}^*) = (t, h, \mathcal{P})$
- 7: **return** (t^*, h^*)

of the best split. This is achieved by changing the equation for the calculation of impurity for supervised PCTs (1). Impurity of a set of examples E (which may contain labeled and unlabeled examples) is calculated as a weighted sum of impurities over the target attribute (Y) and impurities over the descriptive attributes (X_i):

$$Impurity_{SSL}(E) = w \cdot Impurity(E_l, Y) + \frac{1-w}{D} \cdot \sum_{i=1}^D Impurity(E, X_i), \quad (3)$$

where $E = E_l \cup E_u$ is the dataset available at a node of the tree, D is the number of descriptive attributes, X_i is the i^{th} descriptive attribute, and $w \in [0, 1]$ is a weight parameter (which is discussed below). More specifically, the impurity of the target attribute ($Impurity(E_l, Y)$) is calculated using only the labeled examples (E_l), while the impurity over the descriptive attributes ($\sum_{i=1}^D Impurity(E, X_i)$) is calculated by using both labeled and unlabeled examples (E).

The impurity of the target attribute Y over a set of labeled examples E_l is calculated as follows:

$$Impurity(E_l, Y) = \frac{Gini(E_l, Y)}{Gini(E_l^{train}, Y)}, \quad (4)$$

where E_l^{train} represents the labeled part of the *entire* training set.

Differently from the target attribute, which is nominal, the descriptive attributes can be either nominal or numeric. For this reason, we need to separately consider the two cases: if the attribute is nominal as a measure of impurity we use the Gini impurity whereas, if the attribute is numeric, as a measure of impurity we use the variance. More formally, the impurity of a descriptive variable E_i over a set of examples E is calculated as follows:

$$Impurity(E, X_i) = \begin{cases} \frac{Gini(E, X_i)}{Gini(E^{train}, X_i)}, & \text{if } X_i \text{ is nominal} \\ \frac{Var(E, X_i)}{Var(E^{train}, X_i)}, & \text{if } X_i \text{ is numeric} \end{cases} \quad (5)$$

where E^{train} represents the *entire* training set. The variance of the i^{th} attribute is calculated as follows:

$$Var(E, X_i) = \frac{\sum_{j=1}^N (x_i^j)^2 - \frac{1}{N} \cdot \left(\sum_{j=1}^N x_i^j\right)^2}{N}, \quad (6)$$

where x_i^j is the value of the i^{th} target variable of the j^{th} example, and N is the number of examples. The impurity is calculated when evaluating a split for each node of the tree (Table 1, line 4 of the *BestTest* procedure).

Obviously, in order to make the impurity computed on nominal attributes and the impurity computed on numeric attributes comparable, some normalization is necessary. This motivates the denominator of the fractions in (4) and (5) which ensures that all the attributes equally contribute to the calculation of the overall impurity.

Note that, the proposed algorithm is not limited to Gini index or variance as impurity measures. In principle, other impurity measures could be used. For example, the PCT framework also implements the sum of the entropies of class variables, reduced error, gain ratio and m-estimate. We choose to use Gini index because it is one of the most popular measures for impurity of nominal variables. The other obvious choice would be Information gain,

however, Gini index and Information gain perform very similar and mostly it is not possible to decide which of the two measures to prefer (Raileanu and Stoffel 2004). Similarly, for measuring impurity of numeric variables, we choose variance, because variance is *de facto* standard used for regression trees.

The weight parameter w in (3) controls how much the target side or the descriptive side contribute to the calculation of the impurity. Consequently, this controls the amount of supervision in the learning of semi-supervised PCTs. Values of the parameter w close to 1 emphasize more the target attribute, and consequently labeled examples affect the construction of the tree more than unlabeled examples (i.e., there is more supervision). On the other hand, values of w closer to 0 put more emphasis on the descriptive attributes, thus unlabeled examples affect the tree construction more than labeled examples and the tree learning algorithm receives less supervision. The w parameter enables the learning of semi-supervised PCTs to range from fully supervised trees (i.e., $w = 1$) to completely unsupervised trees (i.e., $w = 0$). The ability to control the influence of unlabeled examples with the w parameter is very important, since different datasets may require different amounts of supervision. This aspect is discussed in more detail in Section 4.3.

Henceforth, the semi-supervised predictive clustering trees are denoted as SSL-PCTs, while supervised predictive clustering trees are denoted as PCTs.

2.2 Semi-supervised random forests

Once we have discussed the case of the induction of classical decision trees, we can also extend the semi-supervised learning solution to the case of random forests. A random forest (Breiman 2001) is an ensemble of trees, where diversity among the predictors is obtained by using bootstrap replicates of the training set (as in bagging) and by changing the set of descriptive attributes during learning. Bootstrap samples are obtained by randomly sampling training examples, with replacement, from the original training set, until an number of examples, which is equal to the size of the training set, is sampled. Breiman (1996a) showed that bagging can give notable gains in predictive performance, when applied to unstable learners (for which small changes in the training set result in large changes in the predictions), such as classification and regression tree learners.

A set of descriptive attributes is changed at each node in each decision tree of a random forest: a random subset of the descriptive attributes is taken, and the best attribute is selected from this subset. The number of attributes that are retained is given by a function f of the total number of descriptive attributes D (e.g., $f(D) = \lfloor \log_2(D) + 1 \rfloor$).

In a random forest, the prediction for a new example is obtained by combining the predictions of all trees in the forest. For the classification tasks, different aggregation schemes can be applied, such as majority of probability distribution voting. We use probability distribution voting, as suggested by Bauer and Kohavi (1999). In the probability distribution voting scheme, the base trees predict the probability that an example belongs to each possible class. Thus, each base tree gives its vote (i.e., probability estimate) for each class separately. At the end, the predicted class is the one that has highest sum of probabilities from all base trees.

In Section 2.1 we have discussed a semi-supervised approach to learn classification trees (SSL-PCTs). By using SSL-PCTs, it is possible to build semi-supervised random forests by simply using trees learned with such an algorithm to construct the members of the ensemble, instead of using trees learned with a supervised algorithm. The only difference is that the bootstrap samples are obtained from the whole set of examples E^{train} which includes both labeled and unlabeled examples. We denote the semi-supervised random forest build in such a way as SSL-RF, while supervised random forest is denoted as RF.

3 Experimental design

3.1 Data description

We use 12 binary classification datasets and 12 multi-class classification dataset to evaluate the predictive performance of the proposed methods. The datasets were mainly obtained from the UCI repository (Lichman 2013) and from the OpenML repository (Vanschoren et al. 2014). From these repositories, we selected datasets which differ in the domains they represent, number of attributes and examples, and distribution of classes. We focus on datasets which have more than a thousand of examples, so that we can simulate a scenario relevant for semi-supervised learning, i.e., when large amounts of unlabeled data are available. Namely, in our experimental setup (see Section 3.2), we consider from 25 to 500 labeled examples; therefore, we use datasets with a thousand or more examples to ensure that there are always at least as many unlabeled examples as there are labeled examples.

The characteristics of binary and multi-class classification datasets are given in Tables 2 and 3, respectively.

3.2 Experimental setup and evaluation procedure

We have proposed semi-supervised decision trees (SSL-PCT) and semi-supervised random forest tree ensemble (SSL-RF). As a baseline for comparison, we use the supervised counterparts of these methods, i.e., supervised classification trees (PCT), and supervised random forests (RF). Those are the most reasonable baselines, since the goal is to measure the contribution of the unlabeled data to the overall performance under the same conditions.

As an additional method for comparison, we use the popular semi-supervised algorithm: self-training (Yarowsky 1995). Self-training iteratively uses its own most *confident* predictions on unlabeled examples as an additional training instances. As the base method used in the self-training algorithm we use (supervised) random forest. Confidence of predictions is measured on the basis of votes of an ensemble. For example, if 80% of the trees in ensemble

Table 2 Characteristics of the binary classification datasets

Dataset (Reference)	Domain	N	D/C	P
Abalone (Lichman 2013)	Biology	4177	1/7	50%
Adult (Lichman 2013)	Socio-economic	48842	8/6	76%
Bank (Lichman 2013; Moro et al. 2011)	Economy	4521	9/7	88%
Banknote (Lichman 2013)	Images	1372	0/4	55%
Biodegradation (Lichman 2013; Mansouri et al. 2013)	QSAR	1055	0/41	66%
Diabetes (Lichman 2013)	Medicine	768	0/8	65%
Eyestate (Lichman 2013)	Medicine	14980	0/14	55%
Madelon (Lichman 2013; Guyon et al. 2004)	Synthetic	2000	0/500	50%
Mushroom (Lichman 2013)	Biology	8124	22/0	51%
Pgp (Levatić et al. 2013)	QSAR	932	0/34	52%
Phishing (Lichman 2013)	Security	11055	30/0	55%
Thyroid (Lichman 2013)	Medicine	3772	21/6	93%

N : number of instances, D/C : number of descriptive attributes (discrete/continuous), P : percentage of the majority class

Table 3 Characteristics of the multi-class classification datasets

Dataset (Reference)	Domain	<i>N</i>	<i>D/C</i>	<i>K</i>	<i>P</i>
Baseball (Vanschoren et al. 2014; Simonoff 2013)	Sport	1340	1/15	3	90%
Cardiotocography3 (Lichman 2013)	Medicine	2126	0/35	3	77%
Cardiotocography10 (Lichman 2013)	Medicine	2126	0/35	10	27%
Cmc (Vanschoren et al. 2014)	Socio-economic	1473	7/2	3	42%
Gasdrift (Lichman 2013; Vergara et al. 2012)	Chemistry	13910	0/128	6	21%
Gasdriftdc (Lichman 2013; Vergara et al. 2012)	Chemistry	13910	0/129	6	21%
GesturePhase (Lichman 2013)	Video	9873	0/32	5	29%
Imagesegment (Lichman 2013)	Images	2310	0/18	7	14%
MiceProtein (Higuera et al. 2015)	Biology	1080	3/77	8	14%
Optdigits (Lichman 2013)	Images	5620	0/62	10	10%
Pageblocks (Lichman 2013)	Text	5473	0/10	5	89%
Pendigits (Lichman 2013)	Images	10992	0/16	10	10%

N: number of instances, *D/C*: number of descriptive attributes (discrete/continuous), *K*: number of classes, *P*: percentage of the majority class

predict a positive class for a given instance, confidence of prediction for that instance is set to 0.8. Note that this is a rather straightforward definition of the confidence scores, and more sophisticated variations can be found in the literature (Tanha et al. 2015). In this method used for comparison, the 10% most confident predictions are used as training instances at the next iteration, as proposed by Tanha et al. (2015). As a stopping criteria for self-training, we use the Airbag procedure as proposed by Leistner et al. (2009). This stopping criterion monitors the out-of-bag error (Breiman 1996b) of an ensemble and automatically stops the self-training procedure in the case of predictive performance degradation. Henceforth, we will call this method, used for comparison, SELFTRAININGRF.

In the experiments, both supervised and semi-supervised trees are pruned with the procedure used in C4.5 classification trees (Quinlan 1993). For each variant which uses an ensemble learning approach (i.e., SSL-RF and SELFTRAININGRF), we construct random forests consisting of 100 trees. The trees in random forest are not pruned and the number of random features at each internal node is set to $\lfloor \log_2(D) + 1 \rfloor$, where *D* is the total number of features (following (Breiman 2001)).

Moreover, in order to explore the influence of the amount of labeled data on the predictive performance of the semi-supervised methods and compare them with their supervised counterparts, we execute experiments where we varied the absolute number of labeled examples in the set {25, 50, 100, and 200, 350, 500}. By considering this setup, it is easier to identify the applicability range of a given SSL method with respect to the amount of labeled data. Moreover, this approach is close to a real-world setup: typically, the number of labeled examples is known beforehand, while the number of unlabeled examples can vary.

The main premise of SSL is that a limited amount of labeled data is available, thus it is expected that the success of semi-supervised methods will depend on the amount of labeled data. By varying the amount of labeled data, it is possible to identify for which amounts SSL methods outperform the supervised ones. The alternative to the use of the absolute number of labeled data would be to consider a relative amount of labeled data. However, in such a case, the applicability range becomes a 'moving target', since, for example, 10% of labeled

data can correspond to few tens of examples in one dataset, and a few thousands examples in another dataset.

The labeled data used for training the predictive models (both supervised and semi-supervised) are randomly selected from the available training data, while the remaining examples serve both as unlabeled data and as a test set. Namely, we temporarily remove their labels and provide the examples to the algorithm to serve as unlabeled data during training. When the models are built, the performance is evaluated on the test set composed of the unlabeled examples with their true labels restored. In other words, the evaluation of the semi-supervised methods is performed in the context of transductive learning. For a fair comparison, the supervised methods are trained only on the labeled part of the data and evaluated on the same test set. To get more stable results, the procedure is repeated 10 times with different random initializations. The predictive performance reported in the results is the average of the performance values obtained from the 10 runs.

For each run, we optimize the weight parameter w by performing 3-fold cross-validation on the available labeled part of the training set. During the cross-validation procedure, the semi-supervised methods are supplied with the available unlabeled examples. We consider values of the parameter w ranging from 0 to 1 with a step of 0.1.

To investigate whether the observed differences in performance among the methods are statistically significant, we use the non-parametric Wilcoxon paired signed rank test (Wilcoxon 1945). More specifically, we compare the predictive performance of two methods over multiple datasets. In all the experiments reported in this study, the significance level is set to 0.05. The p-values for the pairwise comparisons are reported, along with an indication of which of the two methods performed better ('-' if the first, '+' if the second performed better). More specifically, when comparing two methods, i.e., Method1 vs. Method2, the '-' sign denotes that the sum of ranks where the first method outperformed the second is higher than the sum of ranks where the second method outperformed the first. The '+' sign denotes the opposite.

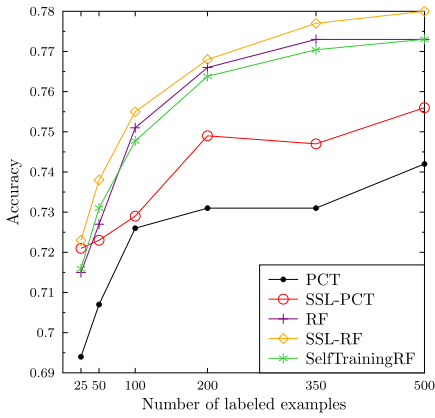
4 Results and discussion

In this section, we present the results of the empirical evaluation. We first present results on binary classification datasets, followed by a presentation of results on multi-class classification datasets. We then investigate the influence of the weight parameter w , which controls the amount of supervision used to learn the models. Next, we analyze the influence of the number of labeled examples on predictive performance of semi-supervised algorithms. Finally, we present examples of supervised and semi-supervised trees and we discuss their interpretability.

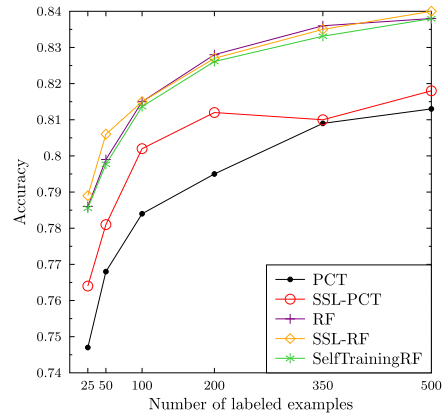
4.1 Binary classification

Figure 1 presents classification accuracy of semi-supervised (SSL-PCT, SSL-RF and SELFTRAININGRF) and supervised methods (PCT and RF) on the 12 binary classification datasets, with an increasing amount of labeled data.

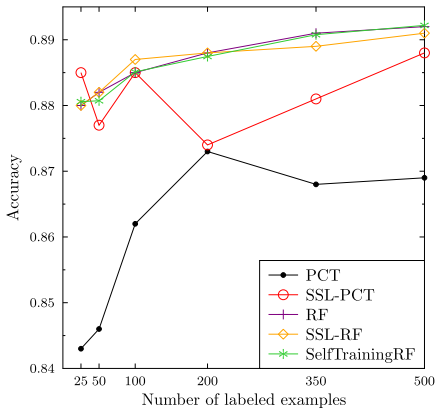
We can observe consistent improvement of SSL-PCTs over supervised PCTs on a number of datasets: Abalone, Adult, Bank, Biodegradation, Diabetes, Madelon and Pgp. The degree of improvement is different from one dataset to another, from relatively small improvement (i.e., around 1% in accuracy) on Biodegradation dataset, to substantial improvement (i.e., up to 16% in accuracy) on Madelon dataset. On other datasets,



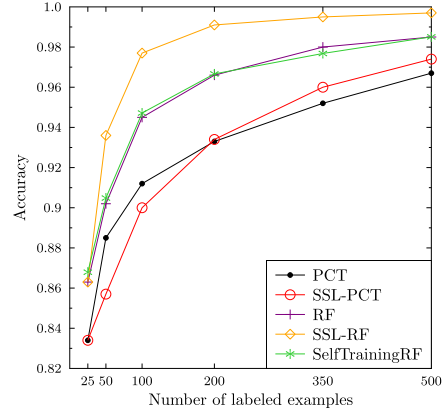
(a) Abalone



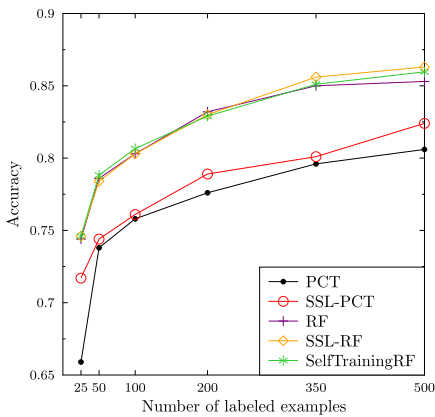
(b) Adult



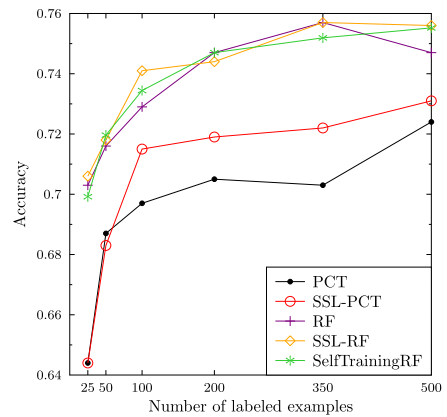
(c) Bank



(d) Banknote

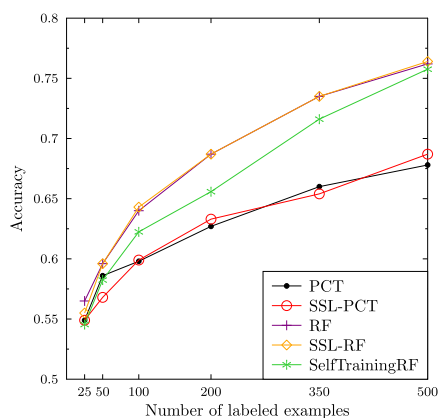


(e) Biodegradation

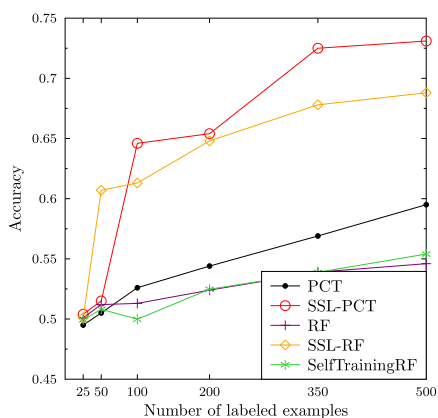


(f) Diabetes

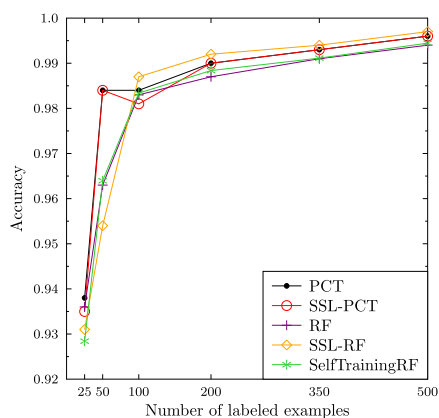
Fig. 1 Accuracy of the supervised and semi-supervised methods on the binary classification datasets



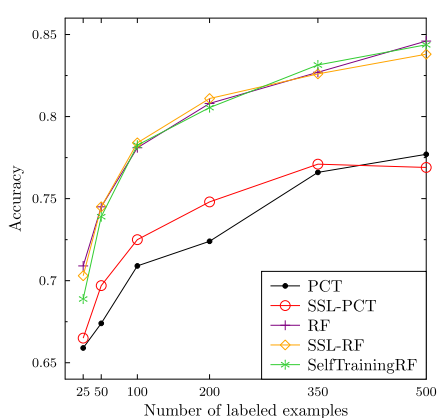
(g) Eyestate



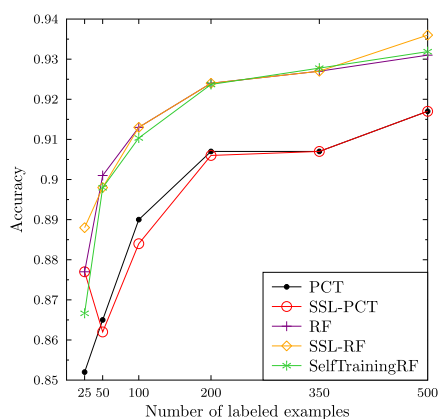
(h) Madelon



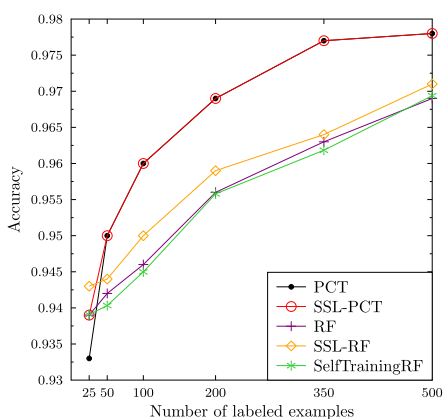
(i) Mushroom



(j) Pgp



(k) Phishing



(l) Thyroid

Fig. 1 (continued)

SSL-PCTs and PCTs seem to show similar behaviour, with small improvements or degradations in performance of SSL-PCT as compared to PCT. The different performance of SSL methods on different datasets is somewhat expected, since success of the semi-supervised methods is known to be domain dependant (Chawla and Karakoulas 2005).

We next compare the performance of semi-supervised random forests (SSL-RF) and supervised random forests (SSL-RF). Consistent improvement of SSL-RF over RF is observed on Abalone, Banknote, Madelon, Thyroid and Mushroom (for 100 and more labeled examples) datasets. It seems that the success of SSL-RF over RF is not directly connected with the success of its base model, i.e., SSL-PCTs. More specifically, improvement of SSL-PCTs over PCTs does not guarantee the improvement of SSL-RF over RF (e.g., Adult and Bank datasets), and vice versa, SSL-RF can improve over RF even if SSL-PCTs does not improve over PCTs (e.g., Banknote and Thyroid datasets). This means that the advantage introduced by the adoption of the semi-supervised learning approach is somehow orthogonal with respect to the advantage introduced by the ensemble learning approach.

The semi-supervised self-training random forest method (SELFTRAININGRF) did not improve notably over RF on any of the datasets. It even consistently degraded the performance of RF on Eyestate dataset. This confirms that self-training suffers from the problem of "reinforcement of mistakes" mentioned before.

Statistical analysis of the performance of the methods (Table 4) shows that SSL-PCTs are better than the PCTs for all amounts of labeled data, while statistical significance is achieved for the smallest amount of labeled data considered (i.e., 25 labeled examples) and when 200 or more of labeled examples are available. Similarly, SSL-RF are better than RF for all amounts of labeled data, and statistically significantly better for 100 and 500 of labeled examples. The proposed semi-supervised random forests are statistically significantly better than SELFTRAININGRF for all amounts of labeled data, except when 50 labeled examples are available.

4.2 Multi-class classification

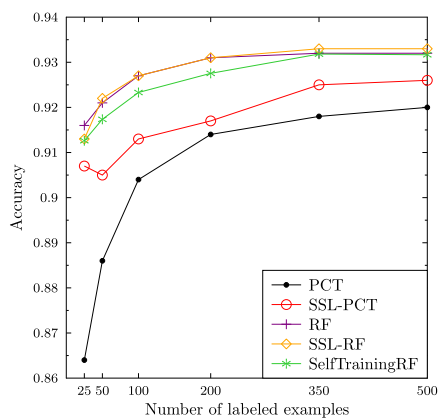
Figure 2 presents classification accuracy of semi-supervised (SSL-PCT, SSL-RF and SELFTRAININGRF) and supervised methods (PCT and RF) on 12 multi-class classification datasets, with increasing amount of labeled data.

Similarly to the case of binary classification, SSL-PCTs consistently improve over the accuracy of PCTs on a number of datasets: Baseball, Cardiotocography3, Cardiotocography10, GesturePhase, Optdigits and Pendigits. On other datasets, the two methods perform very similarly, with the exception of the Cmc dataset, where SSL-PCTs are more accurate than PCTs only when 100 or 200 labeled examples are available.

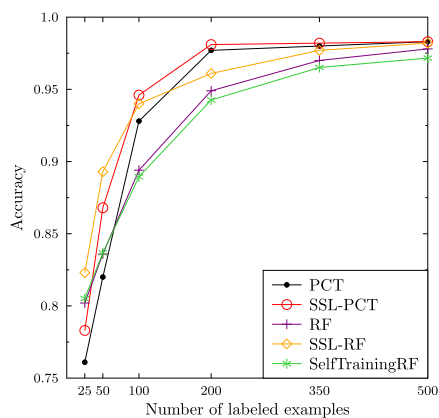
Table 4 *P*-values of the Wilcoxon signed-rank test applied to the performances of the supervised and semi-supervised algorithms on the 12 binary classification datasets considered in this study

Methods	25	50	100	200	350	500
PCT vs. SSL-PCT	0.009 (+)	0.388 (+)	0.066 (+)	0.005 (+)	0.019 (+)	0.019 (+)
RF vs. SSL-RF	0.529 (+)	0.192 (+)	0.002 (+)	0.099 (+)	0.093 (+)	0.012 (+)
SELFTRAININGRF vs. SSL-RF	0.015 (+)	0.072 (+)	0.005 (+)	0.005 (+)	0.015 (+)	0.016 (+)

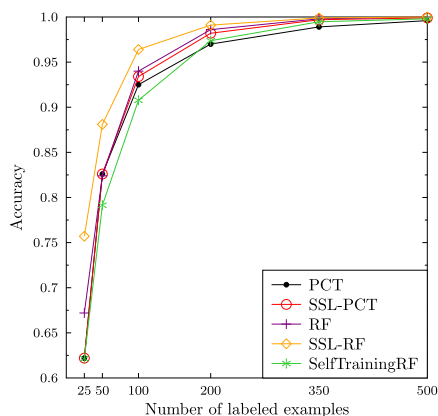
In bold, we report significant *p*-values (< 0.05). The '+' sign means that the second method performs better (SSL-PCT and SSL-RF)



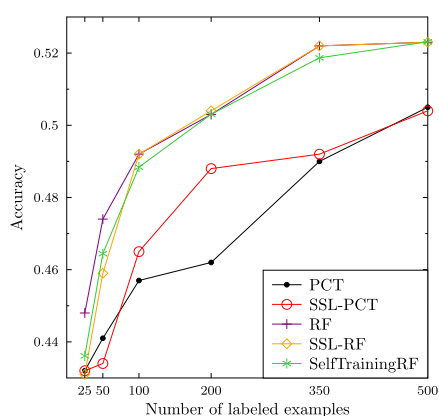
(a) Baseball



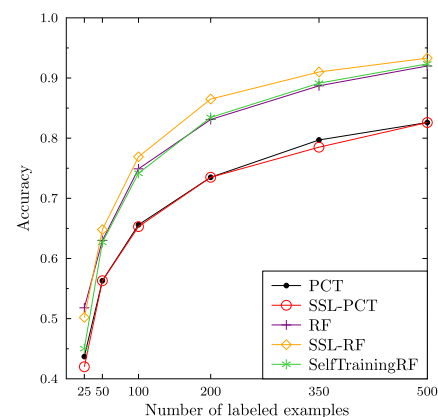
(b) Cardiotocography3



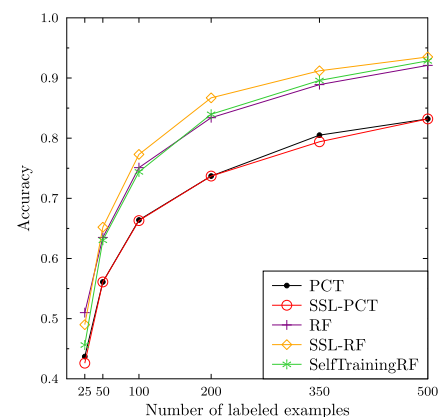
(c) Cardiotocography10



(d) Cmc

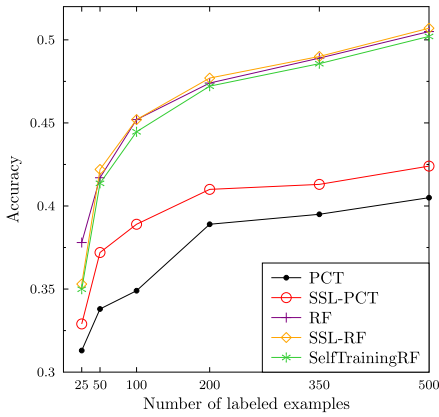


(e) Gasdrift

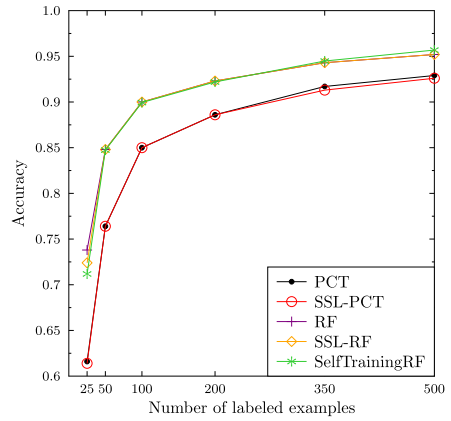


(f) Gasdriftc

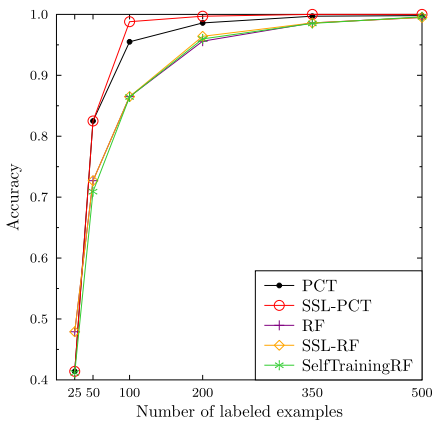
Fig. 2 Accuracy of the supervised and semi-supervised methods on the multi-class classification



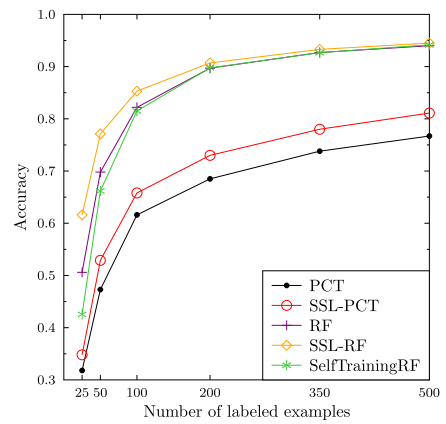
(g) GesturePhase



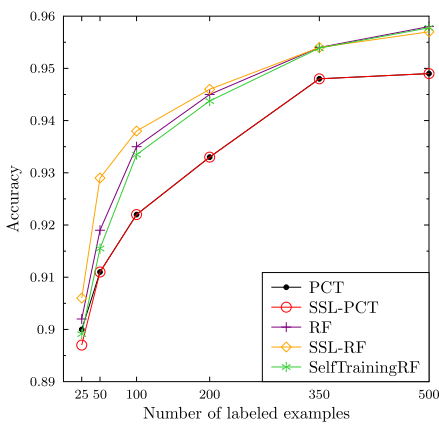
(h) Imagesegment



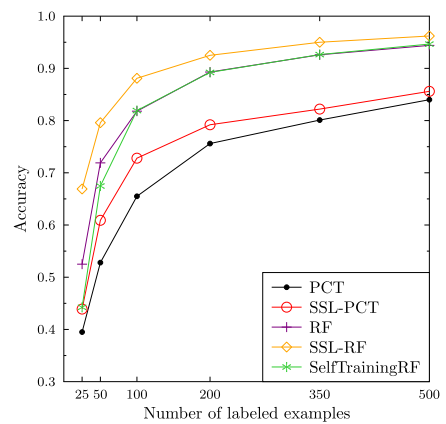
(i) MiceProtein



(j) Optdigits



(k) Pageblocks



(l) Pendigits

Fig. 2 (continued)

SSL-RF consistently improves over RF on Cardiotocography3, Cardiotocography10, Gasdrift, Gasdriftsc, Optdigits, Pageblocks, and Pendigits datasets, while on the other five datasets the performance of the two methods is very similar. Again, the results suggest that SSL-RF can improve over RF regardless of whether SSL-PCTs were able to improve over PCTs or not. For example, SSL-RF was successful on Gasdrift and Gasdriftsc datasets, while SSL-PCT was not. On the other hand, the success of SSL-PCTs does not necessarily transfer to ensemble of SSL-PCTs, as it is the case on Baseball and GesturePhase datasets.

Again, SELFTRAININGRF achieved only very small improvements over RF on few occasions (e.g., Gastrift and Gasdriftsc datasets for 200 or more labeled examples), while more frequently it degraded the performance of the RF. The degradation of performance is not very severe (probably due to the use of the Airbag stopping criterion), however, it happens on the majority of the datasets.

Also in this case we performed a statistical analysis (Table 5) which shows that SSL-PCTs are better than the PCTs for all amounts of labeled data, while statistical significance is achieved for 100 and 200 labeled examples. SSL-RFs are statistically significantly better than RF for all amounts of labeled data, except the smallest amount, i.e., 25 labeled examples. SSL-RFs achieve superior performance with respect to the one of SELFTRAININGRF: statistically significantly better predictive performance is achieved for all the various amounts of labeled data.

4.3 Influence of the w parameter

The w parameter controls the amount of supervision in the induction of the models. Completely unsupervised learning is performed when w is set to 0. Then, as w increases, SSL-PCTs rely more on labeled and less on unlabeled data, arriving at completely supervised learning for $w = 1$.

The ability to fine tune the SSL-PCTs for a given dataset by controlling the amount of influence of unlabeled data is very important in practical applications, since it can protect SSL-PCTs and SSL-RF from severe performance degradation (as compared to their supervised counterparts). When performing semi-supervised learning, there is always a danger (to some extent) of unlabeled data negatively affecting the predictive performances. Several researchers have found that semi-supervised learning may perform worse than supervised learning (Nigam et al. 2000; Cozman et al. 2002; Zhou and Li 2007; Guo et al. 2010). Furthermore, the success of semi-supervised methods was found to be domain dependent, i.e., there are no guaranties that SSL methods will improve the predictive performance of the supervised ones (Chawla and Karakoulas 2005). Moreover, choosing an appropriate SSL

Table 5 P -values of the Wilcoxon signed-rank test applied to the performances of the supervised and semi-supervised algorithms on the 12 multi-class classification datasets considered in this study

Methods	25	50	100	200	350	500
PCT vs. SSL-PCT	0.248 (+)	0.084 (+)	0.014 (+)	0.007 (+)	0.192 (+)	0.081 (+)
RF vs. SSL-RF	0.563 (+)	0.011 (+)	0.011 (+)	0.003 (+)	0.004 (+)	0.02 (+)
SELFTRAININGRF vs. SSL-RF	0.005 (+)	0.003 (+)	0.002 (+)	0.002 (+)	0.006 (+)	0.03 (+)

In bold, we report significant p -values (< 0.05). The ‘+’ sign means that the second method performs better (SSL-PCT and SSL-RF)

method for the problem at hand is still an open question. Thus, even if the main focus in SSL is to outperform the supervised methods, care needs to be taken to make the semi-supervised methods safe, i.e., make sure they do not perform worse than their corresponding supervised methods.

With the w parameter, we provide a safety mechanism for semi-supervised PCTs. In theory, if the optimal value of the parameter w is known, SSL-PCTs and SSL-RF will never perform worse than their supervised counterparts, since PCTs and RF are special cases of SSL-PCT and SSL-RF (respectively) when $w = 1$. However, since the w parameter is optimized via cross-validation on the available labeled data, it is possible that the chosen value of w is not the right one to achieve the optimal test set performance. Thus, in practice, SSL-PCT and SSL-RF can also perform worse than PCT and RF, though this rarely happened in empirical evaluation (Figs. 1 and 2). Considering all the experiments (Figs. 1 and 2), semi-supervised methods (SSL-PCT and SSL-RF) preformed better than their supervised counterparts (PCTs and RF) on 60% of occasions, worse on 17% of occasions, and the same on 23% of occasions. Moreover, the degradation of performance, if it happened, was always for a very small amount as compared to the improvements achieved by SSL-PCT over PCT, and SSL-RF over RF. For example, the average improvement of accuracy on binary classification datasets (Fig. 1) of SSL-PCTs over PCTs is 2.4%, while the average degradation of accuracy is 0.58%.

Figure 3 illustrates the influence of the parameter w on the predictive performance for 3 different datasets. The Optdigits dataset (Fig. 3a) requires little to no supervision, since increasing the value of w leads to worse predictive performance of the SSL-PCT and SSL-RF methods.

For the Baseball dataset (Fig. 3b), the optimal value of w changes with the amount of labeled data, and it is different between the semi-supervised trees and random forests. When small number of labeled examples is available (i.e., up to 50), SSL-PCTs require no supervision and improve a lot over the supervised PCTs. Then, as the number of labeled examples increases beyond 50, improvements are getting smaller and more supervision is needed to achieve the best performance. On the other hand, SSL-RF does not improve notably over the supervised random forests on the Baseball dataset. Generally, larger values of w than at SSL-PCTs are chosen, where SSL-RF perform very similar to RF.

Finally, the Imagesegment dataset (Fig. 3c) is an example of a dataset where the proposed semi-supervised methods do not improve over their supervised counterparts. Regardless of the value of the w parameter, or the amount of the labeled data, SSL-PCT and SSL-RF are not able to improve over supervised learning (except SSL-PCT for 25 labeled examples, however cross-validation narrowly missed the optimal values of the w parameter). Almost always, the value of $w = 1$ is chosen (i.e., supervised learning is performed), and with that, degradation of the performance is avoided.

A general recommendation for the value of w is difficult to provide, since, as shown on Fig. 3, the optimal value of w can vary from dataset to dataset, and even within the same dataset as the amount of labeled data changes. As previously discussed, the performance of semi-supervised methods is known to be domain dependant, i.e., the success of semi-supervised methods can vary from one dataset to another. In other words, different datasets may require different amount of supervision. Since the w parameter controls the amount of supervision, it is not surprising that different values of w are appropriate for different datasets. Hence, this parameter, as done in this empirical evaluation, needs to be optimized by cross-validation for each specific use case.

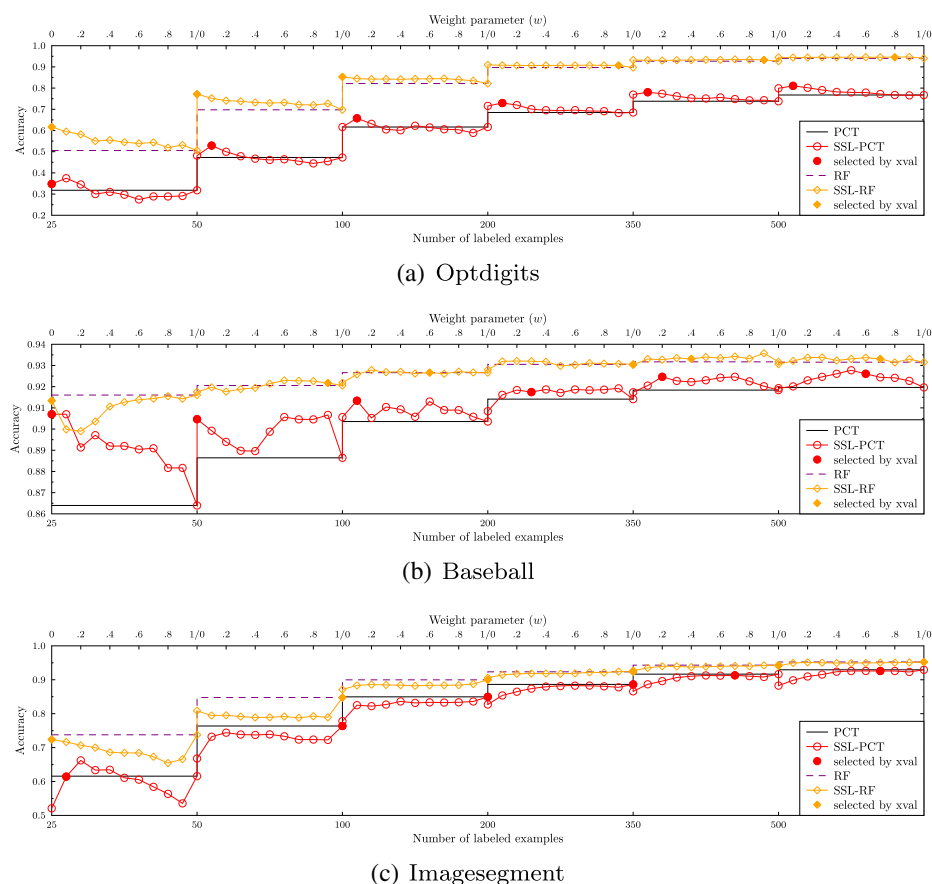


Fig. 3 The effect of the w parameter on the performance of the SSL-PCT (red line) and SSL-RF (orange line) methods on 3 datasets: Optdigits, Baseball and Imagesegment. The values of the w parameter selected by cross-validation and used in the experimental evaluation are denoted with coloured markers

4.4 Influence of the number of labeled examples

The number of available labeled examples is important for the the success of semi-supervised learning. Intuitively, semi-supervised learning has the best potential to improve over supervised learning when a small number of labeled examples is available, while the improvement is expected to decrease as the number of labeled examples increases.

Figure 4 shows that the proposed semi-supervised methods (SSL-PCT and SSL-RF) can improve over their supervised counterparts for any number of labeled examples considered in this study, even for very small amount of labeled data (i.e., 25 labeled examples). However, it seems that for small amounts of labeled data (i.e., up to 50 labeled examples) improvements of semi-supervised methods over supervised methods can vary a lot. For example, semi-supervised random forests on binary classification datasets yield small improvement for 25 labeled examples, while on multi-class classification datasets

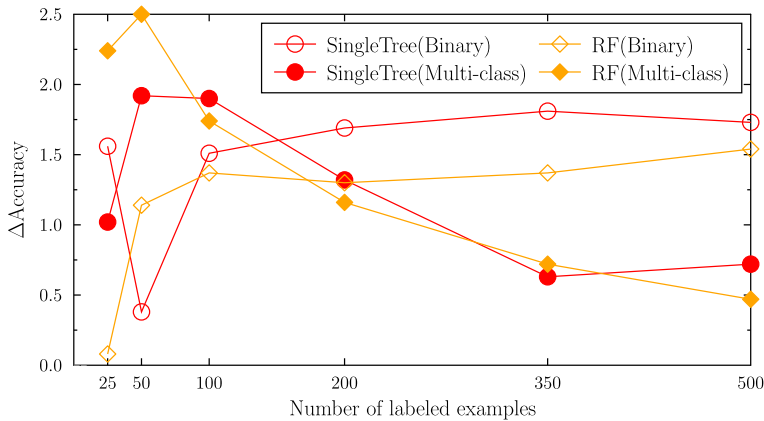


Fig. 4 Average difference in accuracy (Δ Accuracy) across all datasets between semi-supervised SSL-PCT and SSL-RF over their supervised counterparts, PCT and RF, respectively

semi-supervised random forests yield substantial improvement. As the number of labeled examples reaches 100, the degree of improvement somewhat stabilizes, i.e., it does not differ much between single trees and random forests considering binary classification and multi-class classification.

An interesting difference can be observed between binary classification and multi-class classification. As the number of labeled examples is increased beyond 100, the improvement in accuracy of semi-supervised methods on multi-class classification datasets starts to taper off. On the other hand, on binary classification datasets, the improvement is consistent as the number of labeled examples is increasing. This deserves further investigation.

4.5 Interpretability and size of the trees

The models produced by the semi-supervised algorithm are readily interpretable, since they are decision trees. Here, we present examples of decision trees obtained on one binary classification dataset, and one multi-class classification dataset. We discuss the differences between the trees generated by the supervised and the semi-supervised learning algorithms. Our intuition is that if the clustering assumption holds, we can have trees which are less prone to overfitting or, in other terms, smaller trees which better adapt to and predict unseen instances.

From binary classification, we present the trees obtained on the Diabetes dataset (Lichman 2013) with 100 labeled examples (Fig. 5a and b). The task at this dataset is to classify subjects to tested positive for diabetes (class value 1) and tested negative for diabetes (class value 0), on the basis of 8 descriptive attributes: number of times the subject was pregnant, plasma glucose concentration in an oral glucose tolerance test, diastolic blood pressure, triceps skin fold thickness, 2-hour serum insulin, body mass index (*BMI*), diabetes pedigree function (takes into account ancestor's diabetes history), and age. We can observe that the semi-supervised tree (Fig. 5b) is slightly smaller than the supervised tree (Fig. 5a), making it easier to interpret. In spite of its smaller size, the semi-supervised tree is more accurate than the supervised one, meaning that the unlabeled examples contributed to make better splits in the tree. Both trees chose the same split at the root node of the tree: Patients with $BMI \leq 29.85$ are classified as tested negative for diabetes, whereas patients

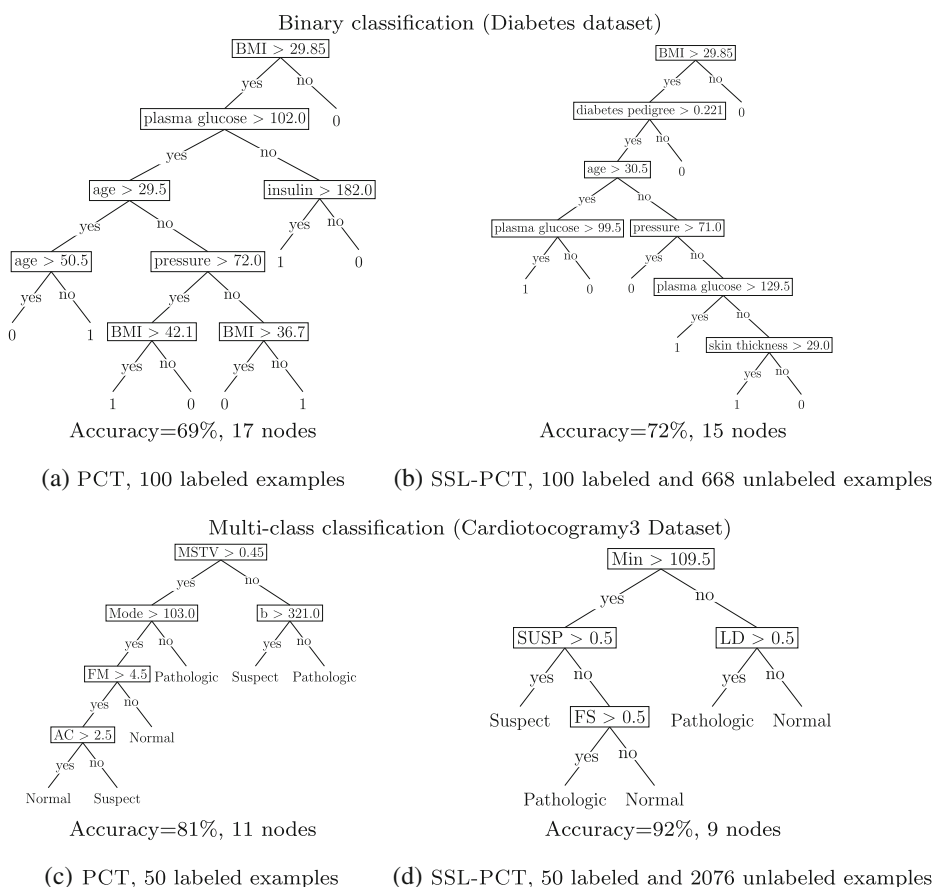


Fig. 5 Examples of trees obtained with the supervised PCT algorithm (**a** and **c**), and the semi-supervised SSL-PCT algorithm (**b** and **d**) on the Diabetes and Cardiotocography3 datasets

with $BMI > 29.85$ are further divided on the basis of other descriptive attributes into positive and negative class. Obesity is known to contribute to increased risk of diabetes, where subjects with $BMI > 30$ are considered obese (Ford 1999). Interestingly, this coincides with the split in the top node of both decision trees. In the lower levels of decision trees, semi-supervised and supervised trees mainly choose different attributes, or different splitting points on the same attribute.

From multi-class classification, we present the trees obtained on the Cardiotocography3 dataset (Lichman 2013) with 50 labeled examples (Fig. 5c and d). This dataset is about classifying the state of the fetus on the basis of cardiotocograms (CTGs) into 3 classes: Normal, Suspect and Pathologic. CTGs are records of measurements of fetal heartbeat and the uterine contractions during pregnancy. From CTGs, 35 features were generated: Various features describing the histogram of foetal heart rate values (Min, Mode, Median etc.), mean value of short term (beat-to-beat) variability (MSTV), heartbeat accelerations (AC), foetal movement (FM), largely decelerative pattern (LD), flat-sinusoidal pattern (FS), etc. We can observe that the tree learned with the semi-supervised algorithm (Fig. 5d) is slightly smaller than the one learned with the supervised algorithm, (Fig. 5c), and that the two trees have

very different structure (i.e., completely different attributes were chosen). The tree learned with the semi-supervised algorithm is much more accurate, meaning that it was able to find better splits.

Tables 6 and 7 present the model sizes of supervised and semi-supervised trees. We can observe that the semi-supervised trees are, almost without exception, smaller than the supervised trees. Recall that, because of the way the impurity measure is defined, semi-supervised trees group examples into clusters that are compact both in the descriptive and the label space, while the supervised trees consider only the label space. Obviously, semi-supervised trees have a more strict clustering criterion, which likely is a reason for the smaller tree size. On average, semi-supervised trees are 25% smaller on binary classification datasets, and 12% smaller on multi-class classification datasets.

Furthermore, Tables 6 and 7 demonstrate that in situations with limited availability of labeled data, semi-supervised learning of PCTs offers better interpretability (i.e., smaller PCTs) and frequently also more accurate trees as compared to supervised learning of PCTs (Figs. 1 and 2). For example, on the Diabetes dataset (Fig. 5a and b), the algorithm for learning supervised PCTs needs approximately 500 labeled examples to achieve predictive performance comparable to the semi-supervised approach with only 100 labeled examples (Fig. 1). Consequently, this can lead to very large, and hence difficult to analyze, supervised trees.

5 Related work

Within SSL, many different methods have been proposed, which are grouped on the basis how they utilize the unlabeled data. Low-density separation methods try to find a labeling

Table 6 Model sizes, in terms of number of nodes, obtained with the supervised algorithm (PCT) and the semi-supervised algorithm (SSL-PCT) on the 12 binary classification datasets considered in this study

Dataset	Number of labeled examples											
	25		50		100		200		350		500	
	PCT	SSL-PCT	PCT	SSL-PCT	PCT	SSL-PCT	PCT	SSL-PCT	PCT	SSL-PCT	PCT	SSL-PCT
Abalone	6.4	4.6	11	4	17.2	9.4	33.4	20	59	39	75.6	52.4
Adult	5.2	2.8	7.6	3.6	16.4	4.8	25.8	6.4	36.8	6.2	54	37.4
Bank	3.2	1	6	1.4	8.4	1	15.2	10.6	25.4	17.4	39.6	3.6
Banknote	5	5	7.6	4.6	11.2	9.2	15.8	13.2	22	18.2	22.6	21.6
Biodegradation	6.4	4.6	10.2	6.6	18.4	14.4	29.6	26.4	48.4	45.2	66.4	64.6
Diabetes	5.4	5.4	10.2	8.4	20.6	14	34.4	31.2	63.6	60	82	92
Eyestate	8.6	8.6	12	11.8	24.6	20.6	49.2	36.8	83.2	63.6	111.2	94.6
Madelon	5.6	6.2	9.2	10	18.2	17	34.2	20.4	60	34.6	82.2	40.8
Mushroom	3.2	3	3.2	3.2	3.4	3.4	4.4	4.4	6.6	6.6	8.4	8.4
Pgp	6.6	6.2	11.8	4.4	19.4	15.4	35.8	29.8	57.8	53.6	74.4	71.2
Phishing	4	3.6	6.8	3.8	8.8	8	13.2	11.8	20.2	20.2	28.8	28.8
Thyroid	2.2	1	2.6	3	4.4	4.4	5.8	5.8	8.6	8.6	9.8	9.8
Average:	5.2	4.3	8.2	5.4	14.3	10.1	24.7	18.1	41.0	31.1	54.6	43.8

Table 7 Model sizes, in terms of number of nodes, obtained with the supervised algorithm (PCT) and the semi-supervised algorithm (SSL-PCT) on the 12 multi-class classification datasets considered in this study

Dataset	Number of labeled examples											
	25		50		100		200		350		500	
	PCT	SSL -PCT	PCT	SSL -PCT	PCT	SSL -PCT	PCT	SSL -PCT	PCT	SSL -PCT	PCT	SSL -PCT
Baseball	3.2	1	4.4	1.8	8	6.4	12	10.8	19	17.2	24	19.8
Cardiotocogramy3	3.6	2.2	6.4	6.8	9.6	11	10.2	10.6	12.2	9.8	14.8	14.8
Cardiotocogramy10	11.6	11.6	15.2	15.2	17.4	18.4	19	20	19	19.4	19	19
Cmc	10.2	10.2	17.8	11.4	31.4	17.8	67	37.4	101.2	68.4	129.8	84.4
Gasdrift	10.6	10.4	15	15	25	22.2	40	40	55.8	56.4	70.8	70.8
Gasdriftdc	10.6	9.6	15	15	24.4	21.6	39.4	39.4	54.4	51.6	70.2	70.2
GesturePhase	10.8	9.8	21	16.2	38	6.8	71.8	66.8	123.2	120.4	172.6	169.8
Imagesegment	12.4	11.2	14.6	14.6	18.4	18.4	23.6	23.6	32.4	31.2	39.2	38
MiceProtein	12.8	12.8	14.8	14.8	15	15	15	15	15	15	15	15
Optdigits	14.8	9.2	20.2	19.2	30.8	31.4	51	46.6	76.4	69.2	92.8	87.6
Pageblocks	3	1.4	4.8	4.8	8.2	8.2	12	9.6	18	18	21.4	18.4
Pendigits	14.8	12.4	20.8	20	28.4	32	43.4	42.4	63.2	55.6	74	72.2
Average:	9.9	8.5	14.2	12.9	21.2	17.4	33.7	30.2	49.2	44.4	62.0	56.7

for the unlabeled data in a way that maximizes the margin of the decision boundary considering both labeled and unlabeled data. Semi-supervised support vector machines are a typical representative of this group (see for instance (Joachims 1999; Ceci 2008)). Finding the exact solution for semi-supervised support vector machines is NP-hard (Zhu 2008), therefore, current implementations, either calculate an approximate solution (Chapelle et al. 2008), or cannot handle more than a few hundred unlabeled examples.

Graph-based methods (Zhang and Wang 2009) use unlabeled data as an additional source of information to infer the structure of the graph. The graph can be considered as a low dimensional representation of the (high dimensional) data. Labels are propagated through the graph assuming label smoothness over the graph. Graph based methods are inherently transductive (i.e., they do not build a model for the whole space, but only infer the labels of the test set). The main problem of these methods is, however, the high computational complexity and recently, some alternative and more efficient solutions were proposed (Liu et al. 2012).

Generative models (Zhu 2008) assume a probabilistic model of the data and use unlabeled, together with labeled data, to estimate the most probable model parameters. Such methods have also been investigated in the multi-relational data mining setting (Malerba et al. 2009; Ceci et al. 2012). The success of generative models depends largely on choosing a probabilistic model which is appropriate for the data.

In this work, we focus on tree-based semi-supervised methods. To the best of our knowledge, semi-supervised classification trees are limited (and interpretable semi-supervised methods in general, as a matter of fact) to the self-training approaches (Yarowsky 1995) wrapped around traditional supervised trees. In an extensive empirical investigation

performed on a large number of UCI datasets, Guo et al. (2010) showed that there is a great risk of performance deterioration when self-training is employed. In fact, the same was shown when the LLGC method (Zhou et al. 2004) was used, and to a somewhat lesser extent when transductive support vector machines (Joachims 1999) were used. Furthermore, they show that "smart" instance selection based on confidence score in self-training and co-training is not necessarily superior to random selection, and that even if unlabeled examples are correctly labeled, improvement of predictive performance is not guaranteed.

Tanha et al. (2015) performed self-training with decision trees and proposed the use of various techniques (such as grafting or Laplace correction) to improve probability estimation in the leaves of the trees. Better probability estimates, used as a proxy for confidence scores, were beneficial for predictive performance of self-trained classification trees. However, even though very similar experimental setup (but not the same) was used by Guo et al. (2010) and Tanha et al. (2015), very different results (serious deterioration of accuracy vs. improvement) were reported on some datasets even if the same method was used. This suggests that even a very small change in the experimental design can have big consequences to the performance of self-training. It is not clear yet, how self-training should be parameterized, in terms of stopping criteria, definition of the confidence score and threshold on the confidence score, to ensure the improvement in performance or at least to prevent a degradation of the predictive performance.

We propose semi-supervised learning of predictive clustering trees, where unlabeled examples are used to improve the quality of the splits of classification trees, i.e., the algorithm does not depend on predicted labels of unlabeled examples. We show that such an approach can substantially improve the accuracy of supervised decision trees, while being reasonably safe to use (i.e., does not reinforce mistakes).

The only previous attempt to use predictive clustering trees in the semi-supervised learning setting is based on the self-training paradigm (Levatic et al. 2014). As previously mentioned, self-training is prone to error propagation and increases the computational complexity of the base method due to the repetitive re-training of the base method. The method presented in Levatic et al. (2014) relies on the intrinsic mechanisms of ensemble learning to estimate the reliability of predictions, thus, it is applicable only to ensemble approaches and cannot produce interpretable models. Furthermore, the machine learning task considered by the method (Levatic et al. 2014) is multi-target regression and the method is not directly applicable to the machine learning tasks considered in this study, i.e., binary and multi-class classification.

As for ensemble-learning solutions, Leistner et al. (2009) proposed to learn semi-supervised random forests where unlabeled data are used to maximize the margin between classes. Similarly as in semi-supervised SVMs, finding the exact solution there is NP-hard, therefore, heuristic techniques are used to find an approximate solution, and the algorithm still depends on predicted labels of unlabeled examples. Liu et al. (2015), proposed semi-supervised random forests, where (similarly to our work) unlabeled data are employed to improve the quality of splits in the trees. They suggest that node splitting is the key issue of tree-based classifiers to achieve high accuracy and avoid overfitting. They use kernel-based density estimation which employs both labeled and unlabeled data to improve categorical probability estimates. These estimates are used directly in impurity measures, such as information gain or Gini index. They show that their method performs favourably to the method proposed by Leistner et al. (2009) on several datasets from the computer vision domain. However, it is not clear if the performance gains observed on random forests would transfer to single trees.

Methodologically related to our work is the work of Blockeel et al. (1998) who proposed clustering trees and suggested that they may be useful when class information is missing. They construct the decision trees in an unsupervised manner, where distances among examples take into account all descriptive attributes. Class labels are not used during the tree construction phase, but only to assign labels at leaf nodes once the tree has been constructed. This concept can be considered as a semi-supervised cluster-then-label approach (Dara et al. 2002; Demiriz et al. 1999; Goldberg et al. 2009). The semi-supervised decision trees that we propose in this work are similar in nature to the ones proposed by Blockeel et al. (1998), but there are several key differences. To construct the trees, we consider simultaneously the descriptive and the target attributes. In other words, clustering and predictive modelling are performed simultaneously, which enables us to fully exploit both the labeled and unlabeled examples. Next, we introduce an additional parameter to control the amount of supervision in the decision trees. This allows us to build fully supervised trees, semi-supervised trees, or fully unsupervised trees, depending on the specific needs of the data mining problem at hand.

6 Conclusions

We proposed a method for semi-supervised learning of classification trees. The trees can be learned from binary and multi-class classification datasets with nominal and/or numeric descriptive attributes. Moreover, the constructed trees can be used as base predictive models in ensembles, such as random forests.

We performed an extensive empirical evaluation showing that the proposed semi-supervised classification trees and ensembles thereof are more accurate than the traditional supervised trees and random forests on many binary and multi-class classification datasets. The attractive predictive performance is not the only advantage of our methods over the existing semi-supervised methods. Quite often, semi-supervised methods promise better accuracy but have high computational demands, many parameters to optimize, or high risk of degrading the performance of their base supervised method, thus limiting their applicability in real-life problems.

In contrast, the methods proposed in this paper are highly useful in practice. First of all, the proposed semi-supervised algorithm for tree learning preserves all of the appealing characteristics of supervised tree learning: semi-supervised trees are fast to learn and are readily interpretable (unlike the models learned by most semi-supervised algorithms). Also, the proposed semi-supervised trees are generally smaller than the supervised ones, making them easier to interpret. Next, due to the parametrization which controls the amount of supervision, the proposed methods are safe to use: They either improve the accuracy of their supervised counterparts, or perform very similar to them.

To conclude, if the user has access to (a large number of) unlabeled examples and the availability of labeled data is limited for the classification task at hand, the methods proposed in this paper will be helpful to improve the predictive performance. Furthermore, if the goal of the modelling is knowledge discovery and understandable models are needed, the proposed semi-supervised trees should be preferred over the traditional supervised trees. On the other hand, if state-of-the art predictive performance is of interest, the proposed semi-supervised random forests are advantageous to traditional supervised random forests.

The proposed approach has potential for development in several different directions in the future. To begin with, we plan to investigate the influence of the class distribution on

the performance of the method. Next, the method can be extended to regression, as well as to more complex machine learning tasks, such as (hierarchical) multi-label classification. Furthermore, the proposed approach could be used to address tasks other than predictive modelling. Namely, with this approach it is possible to perform unsupervised learning, i.e., (hierarchical) clustering while simultaneously providing symbolic descriptions of the clusters. Finally, the method can be used to perform feature ranking for semi-supervised and unsupervised learning by exploiting the principle of feature ranking with random forests.

Acknowledgements We acknowledge the financial support of the Slovenian Research Agency, via the grant P2-0103 and a young researcher grant to the first author, as well as the European Commission, via the grants ICT-2013-612944 MAESTRA and ICT-2013-604102 HBP.

References

- Bauer, E., & Kohavi, R. (1999). An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. *Machine Learning*, 36(1), 105–139.
- Bennett, K., Demiriz, A., & et al. (1999). Semi-supervised support vector machines. *Advances in Neural Information Processing Systems*, 368–374.
- Blockeel, H., De Raedt, L., & Ramon, J. (1998). Top-down induction of clustering trees. In *Proceedings of the 15th Int'l conference on machine learning* (pp. 55–63).
- Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th annual conference on computational learning theory* (pp. 92–100).
- Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2), 123–140.
- Breiman, L. (1996). *Out-of-bag estimation*. Technical report. California: University of California.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32.
- Breiman, L., Friedman, J., Olshen, R., & Stone, C.J. (1984). *Classification and Regression Trees*. Chapman & Hall/CRC.
- Ceci, M. (2008). Hierarchical text categorization in a transductive setting. In *Proceedings of the 8th IEEE international conference on data mining workshops* (pp. 184–191).
- Ceci, M., Appice, A., Viktor, H.L., Malerba, D., Paquet, E., & Guo, H. (2012). Transductive relational classification in the co-training paradigm. In *Proceedings of the 8th international conference on machine learning and data mining in pattern recognition* (pp. 11–25).
- Chapelle, O., Schölkopf, B., & Zien, A. (2006). *Semi-supervised Learning*, vol. 2. MIT Press.
- Chapelle, O., Sindhwani, V., & Keerthi, S.S. (2008). Optimization techniques for semi-supervised support vector machines. *Journal of Machine Learning Research*, 9, 203–233.
- Chawla, N., & Karakoulas, G. (2005). Learning from labeled and unlabeled data: An empirical study across techniques and domains. *Journal of Artificial Intelligence Research*, 23(1), 331–366.
- Cozman, F., Cohen, I., & Cirelo, M. (2002). Unlabeled data can degrade classification performance of generative classifiers. In *Proceedings of the 15th international Florida artificial intelligence research society conference* (pp. 327–331).
- Dara, R., Kremer, S.C., Stacey, D., & et al. (2002). Clustering unlabeled data with SOMs improves classification of labeled real-world data. In *Proc. of the international joint conference on neural networks* (vol. 3, pp. 2237–2242).
- De'ath, G., & Fabricius, K.E. (2000). Classification and regression trees: a powerful yet simple technique for ecological data analysis. *Ecology*, 81(11), 3178–3192.
- Demiriz, A., Bennett, K.P., & Embrechts, M.J. (1999). Semi-supervised clustering using genetic algorithms. In *Proc. of the 5th conference on artificial neural networks in engineering* (pp. 809–814).
- Ford, E.S. (1999). Body mass index, diabetes, and c-reactive protein among us adults. *Diabetes care*, 22(12), 1971–1977.
- Goldberg, A.B., Zhu, X., Singh, A., Xu, Z., & Nowak, R. (2009). Multi-manifold semi-supervised learning. In *Proc. of the 12th international conference on artificial intelligence and statistics* (pp. 169–176).
- Guo, Y., Niu, X., & Zhang, H. (2010). An extensive empirical study on semi-supervised learning. In *Proc. of 10th int'l conf. on data mining* (pp. 186–195).

- Guyon, I., Gunn, S., Ben-Hur, A., & Dror, G. (2004). Result analysis of the NIPS 2003 feature selection challenge. In *Advances in neural information processing systems* (pp. 545–552).
- Higuera, C., Gardiner, K.J., & Cios, K.J. (2015). Self-organizing feature maps identify proteins critical to learning in a mouse model of down syndrome. *PloS one*, 10(6), e0129126.
- Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proc. of the sixteenth international conference on machine learning* (pp. 200–209).
- Kocev, D., Vens, C., Struyf, J., & Džeroski, S. (2013). Tree ensembles for predicting structured outputs. *Pattern Recognition*, 46(3), 817–833.
- Leistner, C., Saffari, A., Santner, J., & Bischof, H. (2009). Semi-supervised random forests. In *Proceedings of the 12th int'l conference on computer vision* (pp. 506–513).
- Levatić, J., Ćurak, J., Kralj, M., Šmuc, T., Osmak, M., & Supek, F. (2013). Accurate models for p-gp drug recognition induced from a cancer cell line cytotoxicity screen. *Journal of Medicinal Chemistry*, 5691–5708.
- Levatic, J., Ceci, M., Kocev, D., & Džeroski, S. (2014). Semi-supervised learning for multi-target regression. In *New frontiers in mining complex patterns - third international workshop, NFMCP 2014, held in conjunction with ECML-PKDD 2014, Nancy, France, September 19, 2014, Revised selected papers* (pp. 3–18).
- Levatić, J., Kocev, D., & Džeroski, S. (2014). The importance of the label hierarchy in hierarchical multi-label classification. *Journal of Intelligent Information Systems*, 1–25.
- Lichman, M. (2013). UCI machine learning repository. <http://archive.ics.uci.edu/ml>.
- Liu, W., Wang, J., & Chang, S.F. (2012). Robust and scalable graph-based semisupervised learning. *Proceedings of the IEEE*, 100(9), 2624–2638.
- Liu, X., Song, M., Tao, D., Liu, Z., Zhang, L., Chen, C., & Bu, J. (2015). Random forest construction with robust semisupervised node splitting. *IEEE Transactions on Image Processing*, 24(1), 471–483.
- Malerba, D., Ceci, M., & Appice, A. (2009). A relational approach to probabilistic classification in a transductive setting. *Engineering Applications of Artificial Intelligence*, 22(1), 109–116.
- Mansouri, K., Ringsted, T., Ballabio, D., Todeschini, R., & Consonni, V. (2013). Quantitative structure–activity relationship models for ready biodegradability of chemicals. *Journal of Chemical Information and Modeling*, 53(4), 867–878.
- Moro, S., Laureano, R., & Cortez, P. (2011). Using data mining for bank direct marketing: An application of the crisp-dm methodology. In *Proc. of the 25th European simulation and modelling conference* (pp. 117–121).
- Nigam, K., McCallum, A.K., Thrun, S., & Mitchell, T. (2000). Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3), 103–134.
- Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. San Francisco: Morgan Kaufmann Publishers Inc.
- Raileanu, L.E., & Stoffel, K. (2004). Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, 41(1), 77–93.
- Rokach, L., & Maimon, O. (2014). *Data Mining with Decision Trees: Theory and Applications*. Series in machine perception and artificial intelligence. World Scientific.
- Simonoff, J.S. (2013). *Analyzing categorical data*. Springer Science & Business Media.
- Slavkov, I., Gjorgjioski, V., Struyf, J., & Džeroski, S. (2010). Finding explained groups of time-course gene expression profiles with predictive clustering trees. *Molecular BioSystems*, 6(4), 729–740.
- Struyf, J., & Džeroski, S. (2006). Constraint based induction of multi-objective regression trees. In *Knowledge discovery in inductive databases, LNCS* (vol. 3933, pp. 222–233).
- Tanha, J., van Someren, M., & Afsarmanesh, H. (2015). Semi-supervised self-training for decision tree classifiers. *International Journal of Machine Learning and Cybernetics*, 1–16.
- Triguero, I., García, S., & Herrera, F. (2015). Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowledge and Information Systems*, 42(2), 245–284.
- Vanschoren, J., Van Rijn, J.N., Bischl, B., & Torgo, L. (2014). Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2), 49–60.
- Vens, C., Struyf, J., Schietgat, L., Džeroski, S., & Blockeel, H. (2008). Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2), 185–214.
- Vergara, A., Vembu, S., Ayhan, T., Ryan, M.A., Homer, M.L., & Huerta, R. (2012). Chemical gas sensor drift compensation using classifier ensembles. *Sensors and Actuators B: Chemical*, 166, 320–329.
- Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 80–83.
- Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on association for computational linguistics* (pp. 189–196).

- Zhang, C., & Wang, F. (2009). Graph-based semi-supervised learning. *Artificial Life and Robotics*, 14(4), 445–448.
- Zhou, Z.H., & Li, M. (2007). Semi-supervised regression with co-training style algorithms. *IEEE Transaction in Knowledge Data Engineering*, 19(11), 1479–1493.
- Zhou, D., Bousquet, O., Lal, T., Weston, J., & Schölkopf, B. (2004). Learning with local and global consistency. *Advances in Neural Information Processing Systems*, 16, 321–328.
- Zhu, X. (2008). *Semi-supervised learning literature survey*. Technical report, Computer Sciences. University of Wisconsin-Madison.