Knowl Inf Syst (2012) 33:631–654
DOI 10.1007/s10115-012-0553-2

REGULAR PAPER

Wrapper positive Bayesian network classifiers

Borja Calvo · Iñaki Inza · Pedro Larrañaga · Jose A. Lozano

Received: 6 December 2011 / Revised: 10 May 2012 / Accepted: 22 August 2012 /
Published online: 9 September 2012

Abstract In the information retrieval framework, there are problems where the goal is to recover objects of a particular class from big sets of unlabelled objects. In some of these problems, only examples from the class we want to recover are available. For such problems, the machine learning community has developed algorithms that are able to learn binary classifiers in the absence of negative examples. Among them, we can find the positive Bayesian network classifiers, algorithms that induce Bayesian network classifiers from positive and unlabelled examples. The main drawback of these algorithms is that they require some previous knowledge about the a priori probability distribution of the class. In this paper, we propose a wrapper approach to tackle the learning when no such information is available, setting this probability at the optimal value in terms of the recovery of positive examples. The evaluation of classifiers in positive unlabelled learning problems is a non-trivial question. We have also worked on this problem, and we have proposed a new guiding metric to be used in the search for the optimal a priori probability of the positive class that we have called the pseudo F. We have empirically tested the proposed metric and the wrapper classifiers on both synthetic and real-life datasets. The results obtained in this empirical comparison show that the wrapper Bayesian network classifiers provide competitive results, particularly when the actual a priori probability of the positive class is high.

Keywords Positive unlabelled learning · Bayesian network classifiers · Wrapper classifiers · Classifier evaluation · Pseudo F

B. Calvo (✉) · I. Inza · J. A. Lozano
Intelligent Systems Group, Department of Computer Science and Artificial Intelligence,
UPV/EHU, Manuel de Lardizabal, 1, 20018 Donostia, Spain
e-mail: borja.calvo@ehu.es

P. Larrañaga
Computational Intelligence Group, Departamento de Inteligencia Artificial,
Universidad Politécnica de Madrid, 28660 Boadilla del Monte, Spain

1 Introduction

The recovery of objects of a particular type from big sets of unlabelled objects is a typical task that can be modelled as a classification problem. The goal is to learn a binary classifier able to distinguish between instances belonging to the class we are interested in (referred to as positive) and instances that do not belong to that class (referred to as negative). From a supervised point of view, we need both positive and negative examples to learn such a classifier but, sometimes, getting negative examples is difficult, expensive or even impossible [6,17,32]. Moreover, even if we can identify some instances that are not positive, we need to be sure that this set of negative examples is representative of all types of negative instances, as having non-representative negative examples may be even harmful for the learning process [3,23].

Semi-supervised algorithms are useful when few labelled examples are available, as they use unlabelled instances to improve the learning process [9]. However, semi-supervised algorithms require labelled examples from all the classes and, thus, they are not able to cope with the absence of negative examples. Given the difficulty (or, even, impossibility) of getting a set of representative negative examples and the unsuitability of (semi-)supervised techniques to handle this absence, one may wonder whether it is possible to learn models from only positive and unlabelled examples.

There are not many works that tackle the theoretical analysis of the learnability from positive examples. Due to its conclusions, Denis [12] is probably the most significant paper. In this work, the author defines the concepts of PAC learnability from positive examples and learnability from positive statistical queries. From this paper, it can be concluded that learning from positive examples is possible, provided that we have some information about the underlying probability distribution of the instances. Part of this information can be estimated from unlabelled examples.

The problem of learning from positive and unlabelled examples has been subject of study for the last ten years. It has been named in the literature partially supervised classification [25], positive example-based learning [36] and positive unlabelled learning (PU learning) [13]. Most of the algorithms designed to learn from positive and unlabelled instances have been developed by the text mining community [13,14,25]. However, during the last years, other problems have also been approached as PU learning, particularly in the computational biology domain [6,8,10,17,31–34,38].

Positive unlabelled learning algorithms can be roughly categorised into three groups. The first group of algorithms considers that all the unlabelled instances are negative examples. This was the approach used before the development of specific PU learning algorithms. In some real-life problems, the ratio of positive examples in the set of unlabelled instances is very low. In such situations, this approach performs well. However, when this ratio increases, this is not a good approach. Some PU learning algorithms exploit the good performance of classifier trained regarding unlabelled instances as negative, combining it with weighting the unlabelled instances or averaging different classifiers. The idea of building and combining different classifiers can be found in Calvo et al. [6] and Sriphaew et al. [30]. The use of weights for the unlabelled instances can be found, for instance, in Elkan et al. [15], Lee et al. [21], Liu et al. [24] and Zhang et al. [37].

In the second group, we have two-step algorithms. These algorithms try to identify (in the set of unlabelled instances) a set of reliable negative examples. Then, the identified negative examples and the positive and (remaining) unlabelled instances are used to build (normally in an iterative way) the classifier. An example of this kind of algorithm is the spy-EM [25] which is based on a modification of the EM algorithm [11]. The spy-EM uses some of the positive cases as 'spies' to identify a set of reliable negative examples and then, in a second step, the EM algorithm is initialised using the identified negative examples and the known positive examples. Other examples of this type of algorithm can be found in Li et al. [22], Pan et al. [27] and Yu et al. [35].

The third category consists of algorithms that directly extract, from the unlabelled instances, information about the underlying probability distribution of the instances and which use this information to help the learning from positive examples. This is a direct application of the conclusions derived from the work of Denis [12]. In this group of algorithms, we can find the positive Bayesian network classifiers (PBCs) [3,4,13] that learn Bayesian network models, such as naive Bayes [26] or tree-augmented naive Bayes (TAN) [16] models, from positive and unlabelled instances.

Positive Bayesian network classifiers use the unlabelled examples to obtain information about the underlying probability distribution of the instances. However, there is a key piece of information that cannot be obtained without negative examples, namely the a priori probability of the positive class.[1] As this probability cannot be properly estimated from positive and unlabelled examples, it remains a parameter of the algorithms.[2] Setting this parameter is the main problem of these algorithms. In Calvo et al. [4], we proposed a Bayesian approach to model the uncertainty about p, providing a more flexible and graphical way to introduce any available knowledge about this probability. However, this Bayesian approach does not solve the problem when no knowledge about p is available.

In this paper, we propose a wrapper approach [19] to set the p_t parameter in PBCs. The idea is to look for the value of p_t that leads to the optimal model in terms of the recovery of positive examples (from now on, we will refer to this value of p_t as the 'optimal' p_t). It is important to point out that this optimal p is not necessarily the actual one. Indeed, the optimal p will depend on the type of model while the actual p is a constant for the problem. As in any other wrapper approach, we need a procedure to evaluate (or at least to compare) the performance of classifiers. In our wrapper approach, we need to evaluate the ability of the models to recover positive examples from the set of unlabelled instances. This is a non-trivial question when no negative examples are available, as most of the commonly used performance measures cannot be directly estimated from positive and unlabelled instances. In this work, we tackle this problem and propose a new metric, the pseudo F measure, that can be used to guide the search for the optimal p_t. Therefore, the contribution of this paper is twofold. On the one hand, we propose a new metric to compare classifiers in the absence of negative examples, and, on the other hand, we use this new metric to guide a procedure to search for the optimal p_t in PBCs.

The rest of the paper is organised as follows: Section 2 introduces the PBCs and presents our wrapper approach to the p_t parameter. Section 3 presents and discusses the results of the evaluation of the wrapper version to the PBCs. Finally, in Sect. 4, some conclusions and guidelines for future work are provided.

[1] Not, at least, directly from positive and unlabelled examples. In Elkan et al. [15], the authors present an interesting theoretical result that could be used to estimate this value under certain assumptions. However, the authors propose other ways to use this theoretical result to learn classifiers, as we will see in the experimental part of the paper.

[2] Henceforth, we will refer to the actual a priori probability of the positive class $P(C = 1)$ as p and, in order to avoid mixing concepts, will refer to the parameter used in the training of PBCs as 'training p' or p_t, to distinguish it from the actual a priori probability of the positive class.

2 Wrapper positive Bayesian network classifiers

The PBCs are algorithms that learn Bayesian network models from positive and unlabelled examples. In the learning process, the user has to provide the algorithm with some information about the a priori probability distribution of the class. In positive naive Bayes (PNB) [13] and positive tree-augmented naive Bayes (PTAN) [3,4], the user has to set this probability (the p_t parameter) by hand. In the averaged version of PNB and PTAN algorithms, APNB and APTAN algorithms [3,4], the uncertainty about p is modelled by means of a Beta distribution whose parameters (α and β) have to be set by the user.

When no information about the a priori probability distribution of the class is available, the user has no means to set either p_t or α and β. In this work, we propose a wrapper version of the PBCs that sets the p_t parameter at its optimal value in terms of the recovery of positive examples.

In this section, the PBCs developed so far will be briefly described, and our wrapper approach to set the p_t parameter will be presented. As the key part of the wrapper approach is the evaluation of the models, this issue will also be elaborated in detail in the following sections.

2.1 Positive Bayesian network classifiers

A naive Bayes model [26] is a Bayesian network classifier based on the assumption that all the predicting variables X_i, $i = 1, \ldots, n$ are conditionally independent given the class variable.[3] Under this assumption, the parameters we have to estimate in a naive Bayes model are the conditional probabilities of each predicting variable given the class value, and the a priori probability of the class, that is, $P(X_i = x_{ij}|C = 1)$, $P(X_i = x_{ij}|C = 0)$ with $i = 1, \ldots, n$ and $j = 1, \ldots, r_i - 1$, and $P(C = 1)$.

The conditional probabilities related with the positive class can be estimated from only positive examples using, for instance, maximum likelihood estimators but, neither the conditional probabilities related with the negative class nor the a priori of the class variable can be estimated without negative examples. However, the former can be put as a function of the latter as follows:

$$P(X_i = x_{ij}|C = 0) = \frac{P(X_i = x_{ij}) - P(X_i = x_{ij}|C = 1)P(C = 1)}{1 - P(C = 1)} \tag{1}$$

where $P(X_i = x_{ij})$ can be estimated from the set of unlabelled examples, as it does not depend on the class. Taking this equation into account, the only probability that cannot be estimated from positive and unlabelled examples is the a priori probability of the positive class. The PNB algorithm [13] uses Eq. (1) to estimate the parameters of naive Bayes models without considering negative examples. Given that the a priori probability of the positive class cannot be estimated from positive and unlabelled examples, it remains as a parameter of the PNB algorithm.[4]

Regardless of the assumption of conditional independence between predicting variables, naive Bayes models can provide good results in many real-life problems. However, its performance in problems where there are strong conditional dependencies between predicting variables can be poor. There are several Bayesian network models that relax the assumption

[3] We are assuming that X_i, $i = 1, \ldots, n$ are discrete variables that can take r_i values and that the class C is a binary variable that can take two values, 0 (or negative) and 1 (or positive).

[4] More details on the estimation of the naive Bayes model parameters in the absence of negative examples can be consulted at Calvo [3] and Denis et al. [13].

of conditional independence in naive Bayes. One such algorithm is the TAN model [16], where the structure of conditional dependencies between predicting variables is represented by a tree (i.e. each predicting variable can have up to one parent besides the class variable). Therefore, TAN models are able to consider conditional dependencies between predicting variables while they keep the number of parameters in check.

In Calvo et al. [4], we proposed an adaptation of the TAN model induction algorithm [16] to the PU learning context, named positive TAN (PTAN). As with PNB, the estimators used in this algorithm depend on p and, thus, the training in PTAN also depends on the p_t parameter, this being the main drawback of these algorithms.

In Calvo et al. [4], we proposed a Bayesian approach to the p_t parameter in PNB and PTAN. In this approach, the uncertainty about p is modelled as a beta probability distribution, and then, the estimators that depend on p are averaged for all the possible values that p can take (from 0 to 1). This Bayesian approach provides a more flexible and graphical way to model the existing knowledge about p, but it does not solve the problem in domains where there is no information about the a priori probability distribution of the class.

2.2 Wrapper approach to the p_t parameter in PBCs

The central idea of our wrapper approach to the learning of PBCs is the search for the value of p_t that optimises the recovery of positive examples. The algorithm proposed progressively refines the search for the optimal p_t reducing, at each step, the size of the interval where it is looked for.

At each step, an interval of search is defined, and a number of models are trained using values of p_t equally distributed within the defined interval (see Fig. 1). Each of the models is evaluated, and the best one is selected as the current optimal one.

At a given iteration i, the limits of the interval are set so that the optimal p_t identified at iteration $i - 1$ is in the middle of the interval[5] (see Fig. 1 for details). The lower limit is set at the highest p_t used in iteration $i - 1$ that is lower than the current optimum. Analogously, the upper limit is set at the lowest p_t used in step $i - 1$ that is greater than the current optimum.

At the first iteration, the interval should cover all the possible values of p_t, from 0 to 1.[6] However, setting p_t at either 1 or 0 can cause problems in the evaluation of the models, as the classifiers trained with these p_t values will label all the instances as positive and negative respectively. To avoid this problem, the limits of the interval at the first iteration should be close (but not equal) to 0 and 1, for example, 0.05 and 0.95.

Figure 2 shows the basic pseudo code of the wrapper PTAN (wPTAN).[7] wPNB is defined in the same way, replacing PTAN algorithm by PNB.

The key part of any wPBC is the evaluation of the models in the recovery task. There are two main aspects of the recovery that we want to maximise, namely the quantity and the quality. The quantity of the recovery can be measured by means of the *recall* (r), which is defined as the ratio of positive examples correctly classified as positive. Conversely, the quality of the recovery is evaluated in terms of the *precision* (p_r), which is defined as the ratio of recovered positive instances that are actually positive.

None of the aforementioned measures alone is enough to evaluate the performance of a classifier in the recovery task. For instance, a classifier that labels all the new instances as

[5] Except when the previous optimum is in an extreme of the interval, in which case the optimum will also be in the corresponding extreme of the new interval, not in the middle.

[6] Obviously, any information about the upper and/or lower bounds of this parameter could be introduced at this point, reducing the initial search space.

[7] Details about the evaluation of the models will be provided in the next section.

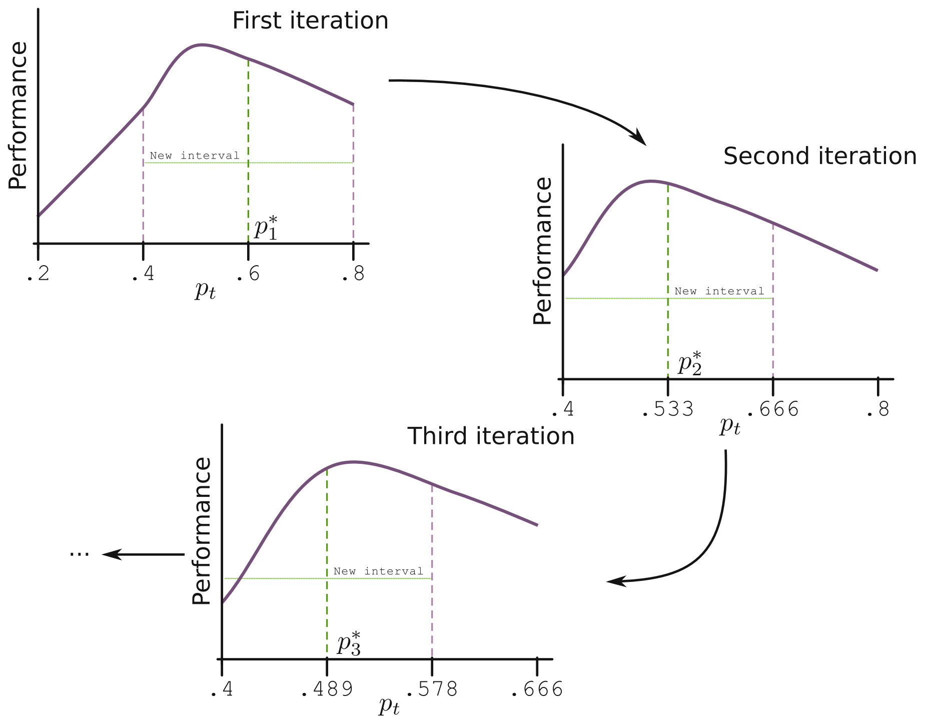


Fig. 1 Diagram of the evolution of the wrapper search for the optimal p_t. p_1^*, p_2^* and p_3^* represents the estimations of the optimal p_t at each iteration

```
Parameters
  $\iota :=$ number of models
  $\kappa :=$ number of rounds
  $p_{min} :=$ minimum value of $p$
  $p_{max} :=$ maximum value of $p$

Repeat $\kappa$ times
  $p_1 = p_{min}$ and $p_\iota = p_{max}$
  For all $i = 2, \cdots, \iota - 1$, $p_i = p_{i-1} + \frac{p_\iota - p_1}{\iota}$
  For all $i = 1, \cdots, \iota$ build a model $\psi_i$ using the PTAN algorithm and $p_i$ as the $p_t$ parameter
  Evaluate the performance of $\psi_i$, $i = 1, \cdots, \iota$ as:
    $F_{ps}^i = \frac{\hat{r}_{p_i}}{\hat{P}(\psi_i(\boldsymbol{X})=1)+p_i}$ where
    $\hat{r}_{p_i}$ is the estimation of the recall of $\psi_i$ and $\hat{P}(\psi_i(\boldsymbol{X}) = 1)$ the estimated probability that
      $\psi_i$ classifies a random instance as positive
  $opt = \arg\max_i F_{ps}^i$
  If $opt \neq 1$
    $p_1 = p_{opt-1}$
  If $opt \neq \iota$
    $p_\iota = p_{opt+1}$

Output
  $\psi_{opt}$ is the wPTAN classifier
```

Fig. 2 Pseudo code of the wrapper PTAN (wPTAN) algorithm

positive would obtain a recall of 1, whereas a classifier that labels just a single (positive) instance as positive would obtain a precision of 1, both classifiers being clearly unsuitable to retrieve positive examples. The quantity and the quality of the recovery are usually combined into a single metric, the *F measure*, which is defined as the weighted harmonic mean of the precision and the recall, and whose most general expression is:

$$\frac{1}{F_\alpha(r, p_r)} = \frac{1}{\alpha + 1}\left(\frac{\alpha}{r} + \frac{1}{p_r}\right)$$
$$\alpha \in [0, +\infty)$$

where α is a weighting factor that allows us to put more emphasis on the quantity (the recall) or the quality (the precision) of the recovery. When the weighting factor α is 1 (i.e. when we regard precision and recall equally important), we have the most commonly used definition of the F measure:

$$\begin{aligned} \frac{1}{F_1(r, p_r)} &= \frac{1}{2}\left(\frac{1}{r} + \frac{1}{p_r}\right) \\ F_1(r, p_r) &= \frac{2rp_r}{r + p_r} = F \end{aligned} \tag{2}$$

From now on, we will use the term F measure, F for short, to refer to $F_1(r, p_r)$.

In our wrapper approach to PBCs, we propose the use of the F measure to evaluate the classifiers in the recovery task. In supervised problems, the F measure can be easily obtained from the confusion matrix, but the absence of negative examples in PU learning problems makes it impossible to directly estimate the precision (and thus, the F measure) from data. In the following section, we will show how the recovery of positive examples can be evaluated in the context of PBCs.

2.3 Evaluation of the recovery of positive examples in PBCs

The evaluation of classifiers in PU learning problems is difficult because of the lack of negative examples. Most of the performance evaluation metrics, including the precision and the recall, are estimated from the confusion matrix that summarises the performance of (binary) classifiers into four values: true positives (TP, the number of positive examples correctly classified), true negatives (TN, the number of negative examples correctly classified), false positives (FP, the number of negative examples classified as positive) and false negatives (FN, the number of positive examples classified as negative).

In PU learning problems, no negative examples are available and, thus, neither TN nor FP can be estimated from data. As a consequence, most of the commonly used performance measures cannot be directly computed.

In the wrapper PBCs, we need to estimate the F measure, which is based on the precision and the recall.

From a probabilistic point of view, the recall of a model ψ is defined as the probability that an actual positive example is classified as positive by ψ, while the precision is the probability that an instance that has been classified as positive by ψ is actually positive:[8]

$$r = P(\psi(\boldsymbol{X}) = 1 | C = 1)$$
$$p_r = P(C = 1 | \psi(\boldsymbol{X}) = 1)$$

where $\psi(\boldsymbol{X})$ represents the class predicted by the classifier ψ and C represents the actual class.

[8] In the most general probabilistic framework, an instance belongs to both the positive and negative class with some probability. In this work, we will focus on the particular case when the probability of belonging to one class is 1 and the probability of belonging to the other class is 0, that is, problems where the instances belong to one, and only one class.

If we take the Bayes rule into account, we have that:

$$p_r = P(C = 1|\psi(\boldsymbol{X}) = 1) = \frac{P(\psi(\boldsymbol{X}) = 1|C = 1)p}{P(\psi(\boldsymbol{X}) = 1)} = \frac{rp}{P(\psi(\boldsymbol{X}) = 1)} \tag{3}$$

Mixing Eqs. (2) and (3), we have that:

$$F = \frac{2rp}{P(\psi(\boldsymbol{X}) = 1) + p} \tag{4}$$

where $P(\psi(\boldsymbol{X}) = 1)$ is the probability that the classifier ψ classifies an instance as positive (i.e. the a priori probability of the positive class in the probability distribution defined by ψ). As we have the classification function ψ, this probability can be obtained classifying all the possible instances and then obtaining the ratio of instances classified as positive.[9]

As we have mentioned before, the recall can be estimated from only positive examples. In this paper, we have used a repeated k-fold cross validation scheme (repeated k-cv) [29] where, at each repetition, the positive examples are divided into k-folds and all the folds except one are used, along with the complete set unlabelled instances, to build a classifier. This classifier is used to predict the class of the instances in the fold which has been left out and the recall is estimated as the ratio of instances classified as positive.[10] The whole process is repeated ρ times, and the final estimation of the recall is calculated as the average of the ρk individual estimations.

Given a training p_t value and a set of positive and unlabelled examples, both PNB and PTAN provide us with a model ψ_{p_t} for which we can compute the recall r_{p_t} and $P(\psi_{p_t}(\boldsymbol{X}) = 1)$. Then, the F measure can be obtained as:

$$F(p_t) = \frac{2r_{p_t}p}{P(\psi_{p_t}(\boldsymbol{X}) = 1) + p} \tag{5}$$

In order to estimate $F(p_t)$, we need to estimate r_{p_t}, $P(\psi_{p_t}(\boldsymbol{X}) = 1)$ and p. As we do not have an estimator of the latter, we can replace it by p_t. This estimation could be used to evaluate the models in our wrapper approach but, given its strong dependence on the p_t parameter, the higher the p_t used in the training the higher the estimated F measure will be, misleading the search towards the highest value for p_t. This undesired behaviour renders Eq. (5) worthless as a guiding metric to optimise p_t.

To overcome this problem, we propose a new metric, the pseudo F (F_{ps}), defined as:

$$F_{ps}(p_t) = \frac{F(p_t)}{2p} = \frac{r_{p_t}}{P(\psi_{p_t}(\boldsymbol{X}) = 1) + p} \tag{6}$$

As p is a constant[11] $F_{ps}(p_t) \propto F(p_t)$ and, thus, $\text{argmax}_{p_t}\{F_{ps}(p_t)\} = \text{argmax}_{p_t}\{F(p_t)\}$. Therefore, instead of using the actual F measure to guide the search for the optimal p_t, we can use the pseudo F.

Therefore, for a given model ψ_{p_t}, the pseudo F measure is estimated as:

$$\hat{F}_{ps}(p_t) = \frac{\hat{r}_{p_t}}{\hat{P}(\psi_{p_t}(\boldsymbol{X}) = 1) + p_t} \tag{7}$$

[9] The feasibility of this calculation will depend on the number of possible instances. When the exact value cannot be obtained, other approaches should be taken.

[10] As the estimation of the recall involves only the positive examples, there is no need for dividing the unlabelled examples into folds.

[11] Note that p refers to the actual a priori probability of the positive class that is a constant for a given problem domain.

where $\hat{r}_{p_t}$ is the estimation of the recall and $\hat{P}(\psi_{p_t}(\boldsymbol{X}) = 1)$ the estimation of the probability that the classifier classifies an instance as positive.

In Lee et al. [21], the authors propose another classifier evaluation metric based on the geometric mean of the precision and the recall. This metric, which we will call here the Lee-Liu metric (L_m), is defined as:

$$L_m = \frac{r \cdot p_r}{p}$$

The Lee-Liu metric is proportional to the square of the geometric mean of the precision and the recall. From Eq. (3), we have that:

$$\frac{P(C = 1|\psi(\boldsymbol{X}) = 1)}{p} = \frac{P(\psi(\boldsymbol{X}) = 1|C = 1)}{P(\psi(\boldsymbol{X}) = 1)}$$
$$\frac{p_r}{p} = \frac{r}{P(\psi(\boldsymbol{X}) = 1)}$$

and thus:

$$L_m = \frac{r \cdot p_r}{p} = \frac{r^2}{P(\psi(\boldsymbol{X}) = 1)} \tag{8}$$

The L_m increases and decreases with the precision and the recall. However, this metric is not proportional to the F measure but to the square of the geometric mean of the precision of the recall and, thus, we cannot use it to guide the search.

3 Experimental evaluation

The evaluation of the wPBCs and the pseudo F presented in this paper has been carried out on both real-life and synthetic datasets. In order to have a proper evaluation of the classifiers, the absence of negative examples in the real-life datasets has been simulated, that is, we actually know the label of the unlabelled samples, but this information is not used in the learning process. A detailed description of how these simulated real-life problems are created can be consulted in Calvo [3] and Calvo et al. [4]. Regarding the synthetic data, they have been sampled from known probability distributions, so that, for any given classifier, we can compute the actual F measure.

The experimental evaluation presented in this section is twofold. On the one hand, we have used the synthetic datasets to evaluate the ability of the pseudo F measure and the learning algorithm to get the optimal p_t. On the other hand, both the synthetic datasets and the real-life ones have been used to compare the performance of wPBCs against other state of the art algorithms.

The rest of the section will present the datasets used in the experimentation, as well as the description and the results obtained in the two experimental frameworks.

3.1 Dataset description

The synthetic datasets have been obtained sampling known probability distributions represented as Bayesian networks. These probability distributions have been randomly generated using the BNGenerator software [18]. Three different random structures were created, setting the maximum number of parents at 2, 3 and 4 respectively. All the structures contain 15 predictive random binary variables. In order to cover a good range of a priori probabilities of the positive class, ten different models were created from each structure, setting the a priori of the positive class at 0.05, 0.15, 0.25, 0.35, 0.45, 0.55, 0.65, 0.75, 0.85 and 0.95 respectively. The rest of the parameters of the Bayesian networks were set at random (sampling a Dirichlet distribution with all the α parameters set at 1).

Each of the models described above was then used to create PU learning problems, consisting of a set of positive examples (drawn fixing the class value at 1) and a set of unlabelled examples (drawn from the original probability distribution and removing the label of the sampled instances). Four different cardinalities for the positive set were used, 100, 500, 1,000 and 5,000 instances. In all the problems, the cardinality of the set of unlabelled examples was set at 5,000. For each model and cardinality of the positive set, fifty different datasets were created, resulting in a total of 6,000 different PU learning problems.

Regarding the simulated problems based on real-life data, they were created based on three supervised databases: ACCDON, Magic[12] and Nursery. The former is the database of splice sites presented in Castelo et al. [7] and the other two are datasets obtained from the UCI repository [1].

Four kinds of data were obtained from ACCDON: acceptor sites coding, acceptor sites intron, donor sites coding and donor sites intron. The cardinality of the sets of positive examples was set at 100, 500 and 1,000, and the size of the set of unlabelled instances was 5,000 in all the problems. The ratio of positive cases used in the sampling process was 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9. In the datasets sampled from Magic database, the cardinalities and ratios used are the same as in ACCDON-based datasets. In this dataset, the class taken as positive is *gamma* (and, thus, *hadron* instances are regarded as negative). In datasets created from Nursery databases, the size of the sets of positive examples was set at 100, 200 and 300, the size of the sets of unlabelled instances at 5,000 and the class used as positive is *spec_prior* (the rest are regarded as negative).[13] The ratios of positive cases in the set of unlabelled instances were the same as in the ACCDON-based problems. As in the case of synthetic data, in all the cases, fifty different sets were drawn for every combination of original dataset, ratio of positive cases and number of positive instances. More details about the construction of these simulated problems can be obtained in Calvo [3].

Table 1 shows a summary of the PU learning problems generated for the experimental part of the paper.

3.2 Evaluation of the pseudo F

The first experimental framework has been designed to check whether the pseudo F is appropriate to estimate the optimal value of the p_t parameter. We have compared the evolution of both the actual F measure and the estimated pseudo F as the p_t parameter changes. The experimentation has only been conducted using the synthetic datasets, as having the original probability distribution allows us to compute the actual F measure. For every set of positive and unlabelled examples, we have run both PTAN and PNB with different values of the p_t, computing in every case, the actual F measure and the estimated pseudo F.

[12] In the original dataset, the attributes are continuous. As the classifiers used work with discrete attributes, the original features have been discretised into three intervals using an equal frequency strategy.

[13] Note that, at most, we need 4,800 positive instances, and only 4,044 are available. To overcome this problem the positive examples in the set of unlabelled instances are resampled when needed (i.e. there are repeated positive examples in the set of unlabelled instances in some problems). This resampling process is used to complete only the unlabelled instances, never the set of known positive examples.

Table 1 Summary of the datasets used in the experimentation

Database	Ratios	C. POS.	C. UNL.	Data
Synthetic data (2 parents)	0.05;0.15;0.25;0.35;0.45	100;500	5,000	2,000
	0.55;0.65;0.75;0.85;0.95	1,000;5,000		
Synthetic data (3 parents)	0.05;0.15;0.25;0.35;0.45	100;500	5,000	2,000
	0.55;0.65;0.75;0.85;0.95	1,000;5,000		
Synthetic data (4 parents)	0.05;0.15;0.25;0.35;0.45	100;500	5,000	2,000
	0.55;0.65;0.75;0.85;0.95	1,000;5,000		
Acceptor sites	0.1;0.2;0.3;0.4;0.5	100;500;1,000	5,000	1,350
Coding	0.6;0.7;0.8;0.9			
Acceptor sites	0.1;0.2;0.3;0.4;0.5	100;500;1,000	5,000	1,350
Intron	0.6;0.7;0.8;0.9			
Donor sites	0.1;0.2;0.3;0.4;0.5	100;500;1,000	5,000	1,350
Coding	0.6;0.7;0.8;0.9			
Donor sites	0.1;0.2;0.3;0.4;0.5	100;500;1,000	5,000	1,350
Intron	0.6;0.7;0.8;0.9			
Magic	0.1;0.2;0.3;0.4;0.5	100;500;1,000	5,000	1,350
	0.6;0.7;0.8;0.9			
Nursery	0.1;0.2;0.3;0.4;0.5	100;500;1,000	5,000	1,350
	0.6;0.7;0.8;0.9			

The first column indicates the database from where the data have been obtained, the second shows the ratios of positive examples in the set of unlabelled instances, the third one is the cardinalities of the set of positive examples, the fourth the cardinality of the unlabelled instances, and the final column is the total number of datasets of positive and unlabelled examples generated (remember that for each combination of cardinalities and ratios, we have generated fifty different datasets)

Figures 3, 4 and 5 show a sample of the results obtained in this part of the experimentation.[14] For every original distribution and every cardinality of the positive set, we have fifty different sets of positive and unlabelled examples, each producing a different plot. To summarise the fifty plots, we have depicted the upper (solid lines) and lower (dashed lines) bounds of both the actual F measure and the pseudo F. In other words, at each p_t, we show the highest and the lowest value obtained by the classifier in the fifty datasets, resulting in two lines that enclose the fifty actual curves.

The results obtained in problems based on data sampled from Bayesian networks with 2- and 4-parents are very similar to those obtained sampling 3-parent models. Therefore, we will only present the results obtained in the latter. However, the complete set of results can be consulted at the supplementary data web page (see footnote 14).

Figure 3 shows the effect of the number of positive examples available. The four plots correspond to the results obtained with PNB in datasets where $p = 0.45$ with an increasing number of positive examples. There are several issues to point out in this figure. First of all, we can see that there is a value of the p_t for which the (actual) F measure is maximum. This is the p_t we have defined as the optimal one (i.e. the value we are looking for). Remember that the optimal and actual p are not necessarily the same. Indeed, the optimal p in the last two plots is slightly higher than the actual one. This can also be observed in the rest of the figures.

[14] The complete set of results can be consulted at the supplementary data web page at http://www.sc.ehu.es/ccwbayes/members/borxa/wPBC/.

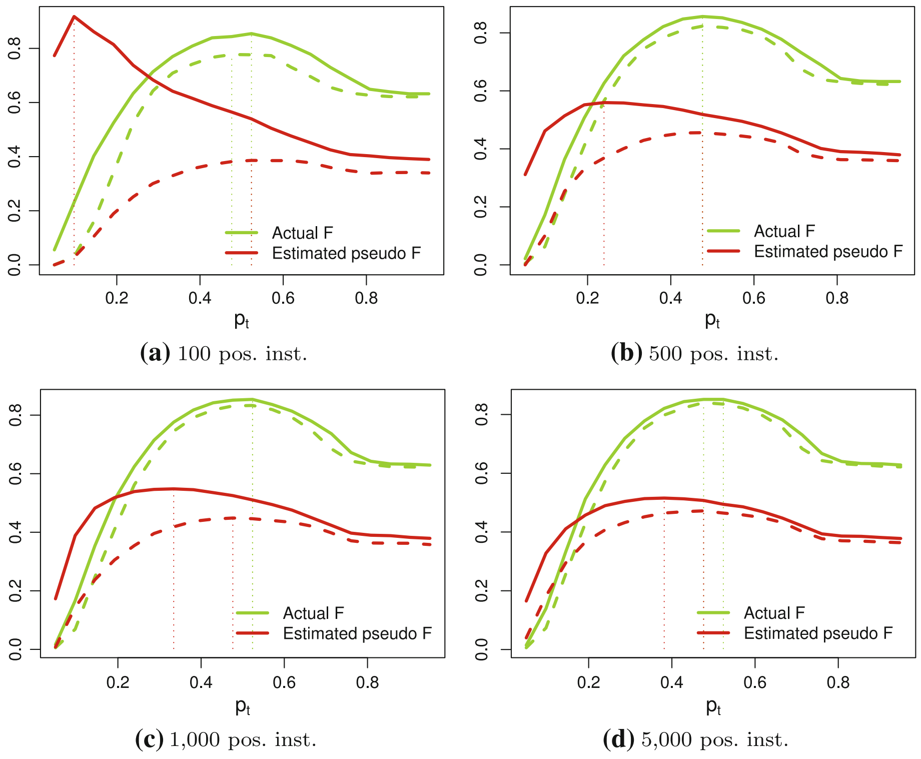
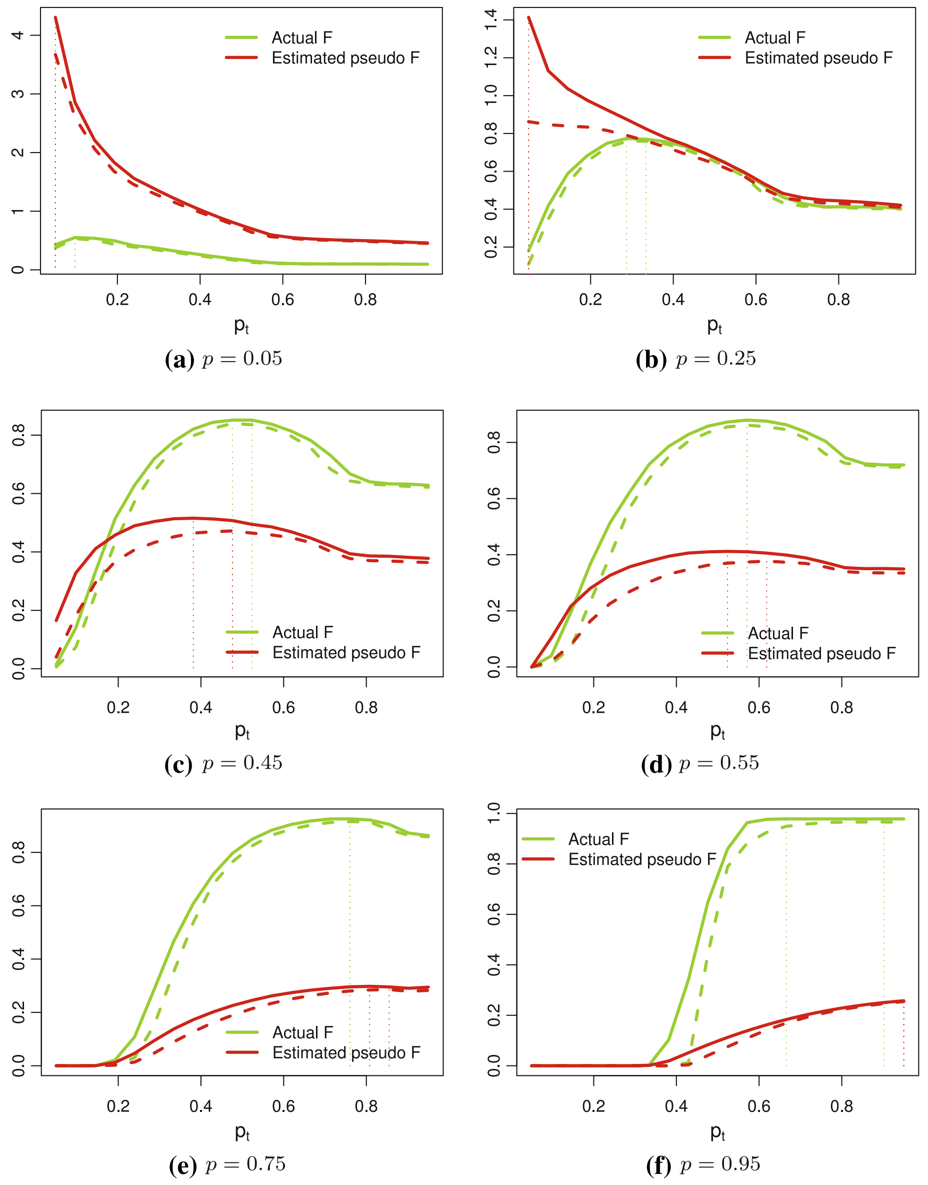


Fig. 3 Examples of the comparison between the actual F measure and the pseudo F measure in synthetic problems sampled from a 3-parent Bayesian network with p set at 0.45. In the four figures, we can see the results obtained when the number of positive examples is set at 100 (**a**), 500 (**b**), 1,000 (**c**) and 5,000 (**d**). In all the cases, the classifier used was PNB. The *solid* and *dashed lines* represent, respectively, the upper and lower bounds of the estimations obtained in fifty repetitions, and the *vertical dotted lines* identify the maximum value in each curve

We can also see that the difference between the upper and lower bound of the metrics also decreases as the number of positive examples increases, especially in the case of the pseudo F. Indeed, we can see that there are huge differences between the upper and lower bounds when we have only 100 positive examples, but these differences are dramatically reduced when 5,000 positive examples are used in the learning process.

Figure 4 shows the evolution of the plot as the actual p changes. We can clearly see in these plots that the optimal p_t increases with the actual p. Nevertheless, we have to point out that the p_t value that maximises the F measure is not necessarily the original p (though in most of the cases it is). We can see this in the last plot, representing the results obtained in problems where the actual p was 0.95 where, for any value of p_t greater than 0.75, there is almost no change in the obtained F measure. We can also see in these plots that, in general, the p_t identified as optimal by the pseudo F is lower than the actual optimal p_t (except for some particular cases when p is close to 1). Finally, the plots show how this bias decreases with p, being close to zero when $p > 0.5$.

Finally, Fig. 5 shows the comparison between the results obtained with PNB and PTAN. We can see in these plots that similar results are obtained with both classifiers (particularly in terms of the optimal p_t and the bias in its identification).



(a) $p = 0.05$ **(b)** $p = 0.25$

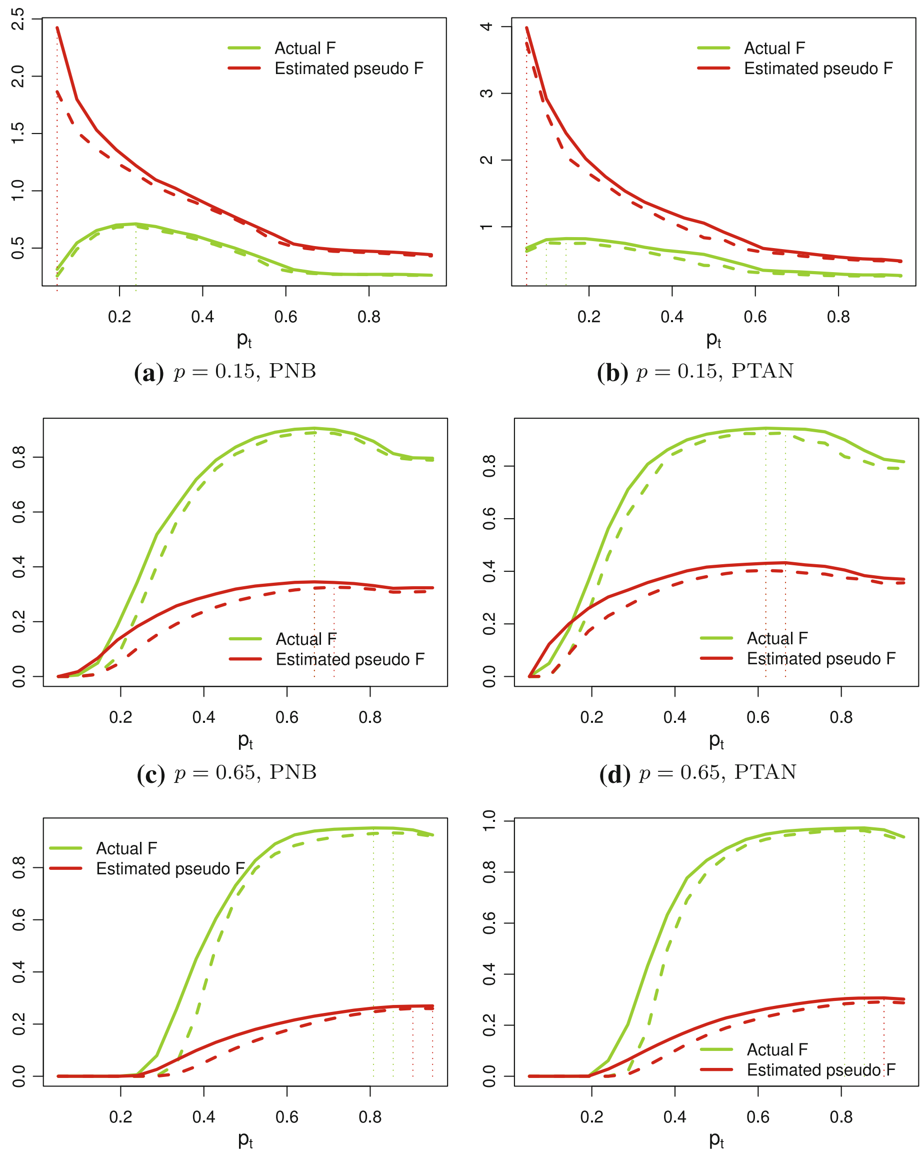
(c) $p = 0.45$ **(d)** $p = 0.55$

(e) $p = 0.75$ **(f)** $p = 0.95$

Fig. 4 Examples of the comparison between the actual F measure and the pseudo F measure in synthetic problems sampled from a 3-parent Bayesian network with different p values. In all the plots, the number of positive examples was 5,000, and the classifier used was PNB. The *solid* and *dashed lines* represent, respectively, the upper and lower bounds of the estimations obtained in fifty repetitions, and the *vertical dotted lines* identify the maximum value in each curve

3.3 Evaluation of the wPBC algorithms

In the second experimental framework, we have compared the performance of the wPBC against other state of the art classification algorithms developed by the PU learning



(a) $p = 0.15$, PNB **(b)** $p = 0.15$, PTAN

(c) $p = 0.65$, PNB **(d)** $p = 0.65$, PTAN

(e) $p = 0.85$, PNB **(f)** $p = 0.85$, PTAN

Fig. 5 Examples of the results obtained when PNB and PTAN are used as the base classifier in synthetic problems sampled from a 3-parent Bayesian network with different p. In all the cases, the number of positive examples was set at 5,000. The *solid* and *dashed lines* represent, respectively, the upper and lower bounds of the estimations obtained in fifty repetitions and the *vertical dotted lines* identify the maximum value in each curve

community. The first group of classifiers used in this comparison is the current PBCs (PNB [13], PTAN, APNB and APTAN [3,4]). As a second group, we have run the spy-EM algorithm presented in Liu et al. [25] using as a base classifier both naive Bayes [26] and TAN [16] models. Finally, we have used the algorithm proposed in Elkan et al. [15] (the weighting unlabelled examples approach) with the same base classifiers as the spy-EM. The list of classifiers run in the comparison is the following:

- *PBC*—Positive Bayesian network classifiers (PNB for naive Bayes-based algorithms and PTAN [4] for TAN-based algorithms). As we are assuming that we have no information about the a priori of the positive class, the p_t has been set at 0.5.
- *APBC*—Averaged positive Bayesian network classifiers (APNB for naive Bayes-based algorithms and APTAN for TAN-based algorithms). In this case, the α and β parameters of the Beta distribution are set at 10, resulting in an average value of p of 0.5 and a variance of approximately 0.01.
- *sEM*—Spy-EM (with naive Bayes and TAN as base classifier).
- *wEN*—Elkan & Noto algorithm (with naive Bayes and TAN as base classifier).
- *wPBCs*—The wrapper version of PBCs proposed in this paper. The estimation of the recall used for computing the pseudo F has been obtained with a 5 times 5-cv. The number of models per iterations was set at 10 and the number of rounds at 5.

These algorithms have been applied to both the real-life and the synthetic datasets described above. The evaluation of the performance of these classifiers has been assessed by means of the F measure, which is, in this framework, a far more informative score than the classification accuracy. The F measure can be computed in our PU learning problems because, as the absence of negative examples is simulated, we actually know the label of the instances in the set of unlabelled examples. The estimation of the F measure has been carried out using a k-cv scheme, with k set at 5. To make the comparison more fair, for each dataset exactly, the same partition of the samples has been used to estimate the performance of all the classifiers.

For a given PU learning problem, we have fifty different sets of positive and unlabelled examples. The results obtained in these datasets have been used to carry out a statistical test to determine whether there are significant differences between the performance of the different algorithms.

Tables 2, 3, 4 and 5 show an extract of the results obtained in synthetic and real-life data-based problems (the complete set of results is available at the supplementary data web page (see footnote 14)). In each row of these tables, the best results and those that show no significant differences with them are highlighted in bold font. The significance of the differences has been determined running statistical tests on the results obtained in the fifty datasets generated for each combination of dataset, p or ratio of positive examples and number of positive examples. The statistical test used to assess the significance of the differences is Wilcoxon's signed rank test.[15] Due to the huge number of statistical tests run, the p values have been corrected using the method presented in Benjamini et al. [2].[16] The p value was set in all the comparisons at 0.01.

Table 2 contains the results obtained in datasets sampled from 3-parent models. The first question to point out in these results is that TAN-based algorithms clearly outperform those based on naive Bayes models. This makes sense, as we know the data come from models where variables are not conditionally independent. Therefore, it is not strange that TAN, with less strong independence assumptions than naive Bayes, provides better results. Within the TAN models, we can see that wEN reports the highest F measures when p is low while wPBC

[15] In a given comparison, we have two paired samples (the evaluation of two classifiers in fifty datasets). The test signed rank test is run on the differences between the performance of the two classifiers in each dataset, in order to exploit the fact that the samples are paired.

[16] Both the tests and the correction were carried out using R. For the multiple testing correction, the package *multtest* was used.

Table 2 Average F measures obtained in the synthetic data-based datasets (maximum number of parents 3)

	p	Naive Bayes based					TAN based				
		PBC	APBC	wPBC	sEM	wEN	PBC	APBC	wPBC	sEM	wEN
100	0.05	15.83	15.45	06.64	09.51	33.02	17.02	24.54	46.53	04.25	**62.53**
	0.15	44.49	40.39	03.11	24.37	33.52	47.44	54.31	25.34	10.82	**60.72**
	0.25	65.03	59.83	15.20	34.56	24.01	**67.15**	**69.89**	08.27	22.41	48.44
	0.35	77.18	72.57	33.48	45.06	15.32	**80.80**	77.79	16.80	32.88	32.50
	0.45	82.28	80.18	57.80	53.87	06.27	**87.34**	85.62	49.32	45.44	17.97
	0.55	83.01	83.50	78.55	58.92	01.05	**88.45**	87.32	71.18	45.52	07.58
	0.65	80.22	80.93	80.34	69.73	00.18	**86.67**	**86.23**	**78.85**	71.24	01.62
	0.75	76.86	77.58	85.42	77.58	00.00	81.80	82.83	**86.98**	82.60	00.00
	0.85	71.11	71.77	**91.38**	82.18	00.00	73.58	74.66	**90.37**	89.79	00.00
	0.95	66.60	67.22	**97.16**	84.23	00.00	67.79	68.96	**96.25**	**93.73**	00.00
500	0.05	15.86	15.21	06.50	09.52	52.59	17.40	24.17	48.03	06.62	**66.66**
	0.15	45.43	40.85	01.86	26.72	59.32	49.59	56.66	26.48	17.46	**71.20**
	0.25	65.94	60.50	13.23	36.06	56.96	70.28	**73.61**	07.26	30.36	69.46
	0.35	78.23	73.00	46.95	46.14	51.32	**81.85**	78.07	09.20	43.02	65.29
	0.45	84.14	82.70	73.82	55.81	42.24	**89.12**	88.09	63.45	57.59	57.20
	0.55	85.31	85.77	83.13	63.91	28.97	**91.19**	**91.06**	88.16	67.82	45.74
	0.65	83.73	84.38	87.59	74.72	12.77	**90.85**	90.53	**91.77**	77.52	29.29
	0.75	80.31	80.93	87.07	83.77	01.81	88.11	88.79	**93.69**	84.84	08.45
	0.85	75.01	75.49	91.97	87.74	00.00	82.36	82.98	**92.47**	91.89	00.23
	0.95	68.67	68.99	97.40	87.69	00.00	72.91	73.32	**97.50**	**97.26**	00.00
1,000	0.05	16.10	15.16	06.27	09.52	53.15	17.84	23.41	48.39	08.19	**64.99**
	0.15	45.42	40.88	01.89	26.07	66.38	50.40	55.18	26.09	22.09	**74.58**
	0.25	66.08	60.88	09.60	38.64	67.61	68.15	**73.94**	06.57	35.23	**75.54**
	0.35	78.44	73.04	50.51	50.39	65.25	**81.04**	77.89	08.93	49.50	74.94
	0.45	84.55	83.86	75.70	61.56	60.22	**89.51**	88.71	73.28	60.01	71.62
	0.55	85.70	86.16	85.42	70.74	52.87	**91.43**	**91.42**	87.35	70.97	65.71
	0.65	84.11	84.76	88.24	78.43	41.27	91.37	91.03	**92.48**	78.79	54.59
	0.75	80.94	81.58	88.12	83.73	22.59	89.12	89.72	**94.65**	85.72	35.23
	0.85	75.76	76.20	91.98	89.59	02.17	84.40	84.87	**93.55**	91.89	09.14
	0.95	70.48	70.73	97.43	91.52	00.03	75.21	75.49	**97.52**	97.46	00.04

(wPTAN) provides the best results when p is high. When p is approximately 0.5, the best results are obtained with (A)PBCs, as we have set the parameters so as the mean p is 0.5.

The performance of wPBC with respect to p is, most likely, related with the ability of the pseudo F to detect the optimal p_t. In particular, we have seen that the bias of the optimum identified by the algorithm with respect to the true value is higher when p is low, resulting in a poor performance of the classifier. Conversely, when p is high, this bias is reduced and, therefore, the performance is improved. This, coupled with the poor performance of other classifiers, renders the wrapper approach the best option. Although in most cases, spy-EM does not outperform wPBCs, we can see that its performance also increases with p.

Table 2 continued

	p	Naive Bayes based					TAN based				
		PBC	APBC	wPBC	sEM	wEN	PBC	APBC	wPBC	sEM	wEN
5,000	0.05	16.24	14.78	05.71	09.52	35.67	18.84	22.37	48.35	09.52	**51.10**
	0.15	45.58	40.97	01.73	26.09	63.50	49.72	55.48	26.24	26.09	**77.25**
	0.25	66.22	60.98	02.27	40.00	74.27	68.65	74.10	07.33	40.00	**82.53**
	0.35	78.41	72.96	50.77	51.85	79.58	80.62	77.95	06.23	51.85	**86.22**
	0.45	84.68	83.57	74.21	62.07	82.65	**89.65**	88.58	67.77	62.07	88.24
	0.55	85.94	86.44	83.97	70.97	84.49	**91.65**	**91.67**	89.64	70.97	89.57
	0.65	84.26	84.92	88.23	78.79	86.07	91.75	91.48	**93.08**	78.79	90.93
	0.75	81.28	81.81	87.46	85.72	88.22	89.79	90.34	**95.16**	85.72	92.24
	0.85	76.68	77.11	92.01	89.28	92.10	85.67	86.12	**96.10**	91.89	94.21
	0.95	71.29	71.47	97.47	93.42	97.39	78.70	78.95	97.64	97.44	**98.14**

In bold font are highlighted the best result and those with no significant differences with the best with a p value of 0.01. Each block in the table corresponds to a given cardinality of the set of positive examples (indicated in the first column)

The performance of wEN algorithm can also easily be understood if we consider how the algorithm works. First of all, this algorithm is based on the fact that, under certain assumptions, the probability of an instance being positive is proportional (with a constant c) to the probability of it being observed [15]. The algorithm exploits this proportionality building a 'nontraditional' classifier trained to distinguish labelled and unlabelled instances. Then the unlabelled instances are 'duplicated', associating with each copy one of the values of the class (i.e. one copy will be a positive example and the other a negative one). These new generated instances are then weighted with a value derived from the estimation of the probability of being observed. Finally, a 'traditional' classifier is trained on the set made of the labelled positive examples and the new generated ones. Obviously, the lower the p, the fewer positive examples, there will be in the unlabelled instances and, thus, the more similar the nontraditional and the traditional classifiers will be (i.e. the closer c will be to 1). Actually, when p is very small, regarding all the unlabelled instances as 'non-positive' would be a quite reasonable approach. However, when p grows, the unlabelled instances are mainly positive and, thus, learning a model that distinguishes between positive and unlabelled instances gets harder, resulting in a poor performance of the whole algorithm.

The results obtained in datasets sampled from 2- and 4-parent models are very similar (see supplementary data web page (see footnote 14) for the complete set of tables).

An extract of the results obtained in real-life datasets can be seen in Tables 3, 4 and 5. The results obtained in these real-life problems confirm what we have observed in synthetic datasets, that is, that the wrapper PBC has the best performance when p is higher than 0.5. We can also see that wEN outperforms other classifiers in most datasets when the ratio is very low, and its performance decreases as p increases. Conversely, the performance of both wPBCs and spy-EM increases with p. In some problems, the spy-EM outperforms wPBC. However, when p is greater than 0.5, wPBCs outperform spy-EM in most of the datasets.

4 Conclusions and future work

Setting the parameter of the current PBCs is, so far, their main drawback. To solve this problem, we have proposed a wrapper approach where the parameter of PBCs (the training

Table 3 Average F measures obtained in problems obtained from acceptor sites intron data

	p	Naive Bayes based					TAN based				
		PBC	APBC	wPBC	sEM	wEN	PBC	APBC	wPBC	sEM	wEN
100	0.1	35.73	35.96	35.54	14.78	**63.45**	33.92	37.75	20.54	20.85	52.03
	0.2	60.39	58.31	19.96	29.90	**63.13**	52.76	57.22	09.27	35.30	42.44
	0.3	**75.94**	74.27	10.86	40.21	57.72	64.08	68.34	09.75	46.54	30.80
	0.4	**84.07**	83.37	18.37	58.33	47.21	71.55	74.42	25.46	56.58	18.42
	0.5	**87.28**	**87.32**	40.60	63.99	31.76	73.54	75.55	40.14	64.86	08.66
	0.6	86.39	**86.93**	69.62	74.45	15.94	73.15	74.42	52.29	70.91	04.90
	0.7	82.91	**83.76**	**81.12**	72.67	03.46	71.09	72.20	**76.69**	76.38	02.33
	0.8	76.66	77.71	**85.63**	**82.84**	00.90	69.22	70.02	78.34	**81.16**	01.54
	0.9	69.19	70.39	**94.92**	**88.10**	00.01	68.06	68.30	94.76	**85.45**	00.79
500	0.1	49.52	48.93	38.62	30.18	75.46	46.46	50.11	25.24	31.29	**78.07**
	0.2	69.51	66.74	22.65	42.86	**77.60**	63.76	67.27	7.94	44.01	**77.17**
	0.3	**81.13**	79.72	09.19	52.62	76.64	75.30	77.58	02.82	54.53	73.51
	0.4	**87.59**	87.06	09.11	61.78	73.74	83.89	84.55	12.93	62.97	66.78
	0.5	**90.31**	**90.36**	41.26	68.40	68.17	87.98	88.53	63.17	70.93	55.20
	0.6	90.40	**90.82**	81.21	77.78	57.83	89.49	89.86	83.44	77.78	41.34
	0.7	88.26	88.88	**92.11**	84.21	40.17	87.18	87.77	86.60	84.22	24.22
	0.8	83.40	84.05	**94.20**	90.02	14.91	81.31	82.19	91.28	89.53	10.63
	0.9	75.44	76.06	95.42	**95.22**	0.56	73.41	74.70	**95.73**	94.74	03.77
1,000	00.10	59.40	59.51	39.17	40.00	80.65	55.85	59.71	25.40	41.94	**84.16**
	0.20	75.31	72.89	22.77	50.00	82.03	68.99	73.18	07.66	50.01	**84.23**
	0.30	**84.35**	83.20	09.27	58.82	82.03	79.61	81.60	01.11	58.83	82.89
	0.40	**89.39**	89.01	10.44	66.67	80.59	87.49	87.26	12.39	66.67	80.03
	0.50	91.38	**91.51**	33.75	73.68	77.56	**91.40**	91.14	73.12	73.68	75.37
	0.60	91.30	91.74	84.28	80.00	72.59	**92.44**	**92.56**	88.94	80.00	68.24
	0.70	89.10	89.66	**93.69**	85.72	62.73	90.72	91.24	91.17	85.72	55.81
	0.80	84.93	85.54	**95.45**	90.98	44.73	86.40	87.16	92.29	90.91	37.43
	0.90	77.46	77.98	95.89	95.84	13.03	77.79	78.66	**96.23**	95.71	17.53

In bold font are highlighted the best result and those with no significant differences with the best with a p value of 0.01. Each block in the table corresponds to a given cardinality of the set of positive examples (indicated in the first column)

p, p_t) is set at the value that maximises the F measure. As in PU learning problems the F measure cannot be directly computed, we have proposed a new metric, the pseudo F, that can be estimated in the context of learning PBCs. We have shown in the experimental part of the paper that this new metric can be successfully used to guide the search for the optimal p_t. In this experimentation, we can also see that the bias of the estimation is higher when the actual p is low, which explains why wPBCs work better when p is high.

We have also carried out an extensive comparison between our proposal and other state of the art algorithms (non-wrapped PBCs [3,4,13], spy-EM [25] and the one of the algorithm presented in Elkan et al. [15], that we have named wEN). These comparisons have been carried out using as base classifiers both naive Bayes and TAN models, in a total of 14,100 datasets. Some of these datasets have been sampled from known probability distributions,

Table 4 Average F measures obtained in problems obtained from magic data

	p	Naive Bayes based					TAN based				
		PBC	APBC	wPBC	sEM	wEN	PBC	APBC	wPBC	sEM	wEN
100	0.1	32.43	29.81	15.14	19.05	13.56	**35.28**	31.36	5.97	19.87	31.73
	0.2	47.99	43.22	22.54	29.93	17.01	**51.31**	47.32	06.05	33.39	19.36
	0.3	59.23	56.94	36.52	38.85	15.99	**63.12**	58.93	25.00	44.86	09.85
	0.4	65.00	65.10	52.13	45.98	09.55	**68.37**	66.92	38.30	55.28	03.93
	0.5	65.97	66.45	53.95	52.92	14.04	70.41	**71.97**	61.36	64.42	01.70
	0.6	68.06	68.50	**67.85**	56.04	07.37	69.31	**74.42**	**71.71**	71.92	00.77
	0.7	67.18	67.79	**74.59**	61.09	08.10	67.68	75.29	**77.91**	77.58	00.59
	0.8	65.88	66.30	**82.97**	64.25	04.07	64.03	76.20	**85.01**	82.73	00.46
	0.9	65.09	65.52	**84.94**	67.14	05.63	64.02	76.93	**93.80**	86.68	00.18
500	0.1	43.86	40.71	09.07	27.81	36.91	48.70	44.17	04.36	30.16	**54.70**
	0.2	55.93	51.04	21.27	36.52	40.12	**61.49**	56.66	05.72	42.24	48.90
	0.3	65.61	62.27	46.76	44.14	34.56	**70.42**	66.38	31.68	52.01	41.67
	0.4	69.41	69.73	48.99	50.93	36.03	**75.08**	72.79	60.16	61.08	30.44
	0.5	72.41	72.86	67.47	55.88	31.02	76.95	**77.59**	**76.22**	68.78	15.89
	0.6	72.48	72.92	75.74	61.34	20.95	76.01	**79.75**	**80.18**	76.90	05.50
	0.7	71.87	72.26	82.70	65.13	16.54	74.39	80.97	**84.38**	82.48	02.01
	0.8	68.93	69.22	88.32	68.62	13.58	71.66	80.24	**90.03**	87.75	00.63
	0.9	67.63	67.83	92.35	69.18	07.92	68.22	81.04	**95.14**	90.63	00.33
1,000	0.1	52.14	47.30	01.77	34.97	47.15	58.57	54.21	04.60	39.65	**62.47**
	0.2	61.59	55.84	20.74	43.26	47.94	**66.88**	63.67	02.23	49.89	60.91
	0.3	69.41	66.24	47.57	49.94	48.51	**73.76**	70.60	38.61	59.05	56.26
	0.4	73.61	73.83	65.31	55.07	47.50	**77.52**	76.09	67.33	66.25	49.56
	0.5	74.16	74.63	71.29	59.41	46.94	78.53	**79.62**	**75.82**	72.93	40.57
	0.6	75.66	76.14	81.26	65.91	37.82	78.75	81.82	**83.68**	79.24	26.67
	0.7	73.05	73.40	84.53	70.17	34.21	76.52	81.85	**86.19**	84.18	12.12
	0.8	72.45	72.70	**90.64**	70.43	20.49	74.44	81.93	**90.96**	89.74	03.88
	0.9	69.06	69.26	93.94	71.48	17.63	70.89	81.87	**95.56**	**94.14**	01.28

In bold font are highlighted the best result and those with no significant differences with the best with a p value of 0.01. Each block in the table corresponds to a given cardinality of the set of positive examples (indicated in the first column)

allowing us to control the parameters in the datasets and to calculate the actual F measure and pseudo F. The rest of the datasets have been generated from different supervised datasets, simulating the absence of negative examples.

The main conclusion we can draw from this experimentation is that there is not a best classifier for the whole range of p. When this value is low, wEN is the algorithm that performs the best, but as we approach to $p = 0.5$ non-wrapper PBCs outperform the rest (mainly because, in the absence of any information about p, we set the parameters so as the mean p is 0.5). However, as p grows, the wrapper approach presented here increases its performance, outperforming the rest of the classifiers (including spy-EM, whose performance also increases with p). Moreover, in some problems (those derived from the Nursery dataset), wPBCs provide competitive results even though $p \leq 0.5$.

Table 5 Average F measures obtained in problems obtained from nursery data

	p	Naive Bayes based					TAN based				
		PBC	APBC	wPBC	sEM	wEN	PBC	APBC	wPBC	sEM	wEN
100	0.1	35.41	**47.98**	35.93	21.05	00.12	38.56	**47.88**	03.66	21.05	09.23
	0.2	58.10	**66.56**	**50.00**	35.48	00.01	59.79	**64.43**	38.36	35.48	0.23
	0.3	**75.34**	**76.06**	69.90	47.76	00.00	70.16	**74.47**	55.28	47.76	00.00
	0.4	**83.29**	**83.10**	77.58	58.33	00.00	76.29	77.95	59.99	58.33	00.00
	0.5	**86.39**	**86.46**	84.20	67.53	00.00	79.01	79.91	70.91	67.53	00.00
	0.6	85.09	**85.53**	**85.15**	75.61	00.00	77.73	78.91	75.38	75.61	00.00
	0.7	81.43	81.98	**83.57**	82.76	00.00	74.28	75.57	**83.66**	82.76	00.00
	0.8	76.48	77.08	**89.51**	89.14	00.00	70.03	71.21	**89.61**	89.14	00.00
	0.9	70.58	71.28	**93.11**	94.41	00.00	65.61	67.20	92.85	**95.07**	00.01
200	0.1	40.57	54.07	38.71	23.73	3.75	44.40	**60.94**	01.28	23.73	24.28
	0.2	59.57	70.26	**58.38**	37.50	01.06	62.07	**73.69**	39.70	37.50	03.28
	0.3	76.65	77.78	73.55	49.28	00.15	73.91	**80.25**	70.86	49.28	00.24
	0.4	**86.93**	86.35	**84.28**	59.46	0.00	81.59	**86.70**	76.89	59.46	00.01
	0.5	**89.59**	**89.66**	88.25	68.35	0.00	85.25	87.37	82.94	68.35	00.00
	0.6	**88.75**	**89.13**	**89.55**	76.19	00.00	85.12	85.77	83.07	76.19	00.00
	0.7	**85.47**	**85.94**	**87.32**	83.15	00.00	80.99	81.91	84.68	83.15	00.00
	0.8	80.36	80.85	89.89	89.37	00.00	76.09	77.14	**90.26**	89.37	00.00
	0.9	73.73	74.14	**95.17**	**95.07**	00.00	69.92	70.89	**95.20**	94.96	00.00
300	00.1	43.95	57.39	41.74	26.23	12.24	47.39	**64.67**	01.22	26.23	36.91
	0.2	61.77	73.69	54.34	39.39	06.57	65.68	**77.65**	38.40	39.39	12.71
	0.3	76.77	80.76	69.68	50.70	02.14	75.38	**83.93**	71.18	50.70	02.19
	0.4	87.68	87.00	85.92	60.53	00.12	84.30	**89.22**	82.37	60.53	00.19
	0.5	**90.61**	**90.79**	88.90	69.14	00.00	87.38	**90.50**	85.38	69.14	00.02
	0.6	90.79	**91.16**	**90.54**	76.74	00.00	87.48	88.54	86.15	76.74	00.01
	0.7	87.48	87.89	**90.14**	83.52	00.00	84.63	85.19	86.12	83.52	00.00
	0.8	82.61	83.02	90.19	89.59	00.00	79.19	79.89	**90.76**	89.59	00.00
	0.9	75.46	75.81	95.36	95.08	00.00	72.45	73.34	**95.60**	95.06	00.00

In bold font are highlighted the best result and those with no significant differences with the best with a p value of 0.01. Each block in the table corresponds to a given cardinality of the set of positive examples (indicated in the first column)

We can, therefore, conclude that the wrapper positive Bayesian network classifiers introduced in this work provide competitive results, particularly when the actual probability of the positive class is high.

Regarding future work, it would be interesting to explore the possibility of modifying the algorithm presented here to improve its performance when p is low, which is the type of scenario where the wrapper approach provides the worst results. In this line, it would be advisable to analyse and integrate the ideas presented in Elkan et al. [15], as they show a good performance in this context. The good performance of this algorithm comes, most likely, from the fact that regarding the unlabelled instances as belonging to the negative class are a good approach when the ratio of positive examples in this set is low.[17] Therefore, in future developments, we can explore how this assumption can be exploited in the search for the optimal p_t. In the same line, it would also be interesting to analyse the pseudo F measure and its estimation from a theoretical point of view, in order to identify the origin of the bias in the estimated p_t when the actual p is low which, at the end of the day, is the reason for the bad performance of the wrapper approach when the ratio of positive examples is low. Following with the analysis of the pseudo F measure, another interesting research line would be analysing the use of this metric in other contexts, such as the evaluation of classifiers in real-life problems. In this work, we have pointed out that the p value that maximises the F measure is not necessarily the actual one. A theoretical and experimental analysis of the optimal p would help understanding whether it can be regarded as an estimation of p and, if so, under which conditions.

Another line for future development is the application of the wrapper concept in other aspects of the data mining pipeline, implementing algorithms guided by the pseudo F measure. In this sense, there are two lines to follow. The first one has to do with the feature subset selection [5], where we can develop algorithms that select the subset of variables that leads to the best performance. The second one has to do with the classifier learning process itself, where, in some cases, wrapper approaches can be used to get the model with the best performance. Two examples of algorithms from the Bayesian network literature that could be adapted to the PU learning context are the semi naive Bayes and the selective naive Bayes learning algorithms [19,20,28].

Acknowledgments This work has been partially supported by the Saiotek and Research Groups 2007–2012 (IT-242-07) programs (Basque Government), TIN2010-14931, TIN2010-20900-C04-04 Consolider Ingenio 2010-CSD2007-00018 and Cajal Blue Brain Project (C080020-09) projects (Spanish Ministry of Science and Innovation), the COMBIOMED network in computational bio-medicine (Carlos III Health Institute) and the Diputación Foral de Gipuzkoa (2011-CIEN-000060-01).

References

1. Asuncion A, Newman DJ (2007) UCI machine learning repository
2. Benjamini Y, Yekutieli D (2001) The control of the false discovery rate in multiple testing under dependency. Ann. Stat 29(4):1165–1188
3. Calvo B (2008) Positive unlabelled learning with applications in computational biology. Ph.D. thesis, Facultad de Informatica–UPV/EHU
4. Calvo B, Larrañaga P, Lozano JA (2007a) Learning Bayesian classifiers from positive and unlabeled examples. Pattern Recognit Lett 28:2375–2384
5. Calvo B, Larrañaga P, Lozano JA (2009) Feature subset selection from positive and unlabelled examples. Pattern Recognit Lett 30(11):1027–1036
6. Calvo B, López-Bigas N, Furney SJ, Larrañaga P, Lozano JA (2007b) A partially supervised classification approach to dominant and recessive human disease gene prediction. Comput Methods Program Biomed 85:229–237
7. Castelo R, Guigó R (2004) Splice site identification by idlbns. Bioinformatics 4(20):i69–i76
8. Cerulo L, Elkan C, Ceccarelli M (2010) Learning gene regulatory networks from only positive and unlabeled data. BMC Bioinform 11:228
9. Chapelle O, Zien A, Schökopf B (2006) Semi-supervised learning. MIT Press, Cambridge
10. Chen Y, Li Z, Wang X, Feng J, Hu X (2010) Predicting gene function using few positive examples and unlabeled ones. BMC Genomics 11(Suppl 2):S11

[17] The algorithm does not assume this, but it builds a classifier to differentiate labelled and unlabelled examples which, in practical terms, is the same as building a classifier to distinguish positive and negative examples assuming that all the unlabelled instances are negative.

11. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Ser B 39:1–38
12. Denis F (1998) PAC learning from positive statistical queries. In: Proceedings of the 9th international conference on algorithmic learning theory. Springer, New York, pp 112–126
13. Denis F, Gilleron R, Tommasi M (2002) Text classification from positive and unlabeled examples. In: Proceedings of the 9th international conference on information processing and management of uncertainty in knowledge-based Systems. IPMU 2002, pp 1927–1934
14. Duan Q, Miao D, Jin K (2007) A rough set approach to classifying web page without negative examples. In: Proceedings of the 11th Pacific-Asia conference on advances in knowledge discovery and data mining. Springer, New York, pp 481–488
15. Elkan C, Noto K (2008) Learning classifiers from only positive and unlabeled data. In: Proceeding of the 14th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, USA, pp 213–220
16. Friedman N, Geiger D, Goldszmit M (1997) Bayesian network classifiers. Mach Learn 29:131–163
17. Furney SJ, Calvo B, Larrañaga P, Lozano JA, Lopez-Bigas N (2008) Prioritization of candidate cancer genes-an aid to oncogenomic studies. Nucleic Acids Res 36(18):e115
18. Ide JS, Cozman FG (2002) Random generation of Bayesian networks. In: Proceedings of the 16th Brazilian symposium on artificial intelligence: advances in artificial intelligence. Springer, New York, pp 366–375
19. Kohavi R, John GH (1997) Wrappers for feature subset selection. Artif Intell 97(1–2):273–324
20. Kononenko I (1991) Semi-naive Bayes classifiers. In: Proceedings of the 6th European working session on, learning, pp 206–219
21. Lee WS, Liu B (2003) Learning with positive and unlabeled examples using weighted logistic regression. In: Proceedings of the twentieth international conference on machine learning, pp 448–455
22. Li X, Liu B (2003) Learning to classify texts using positive and unlabeled data. In: Proceedings of the eighteenth international joint conference on artificial intelligence 2003, pp 587–594
23. Li X-L, Liu B, Ng S-K (2010) Negative training data can be harmful to text classification. In: Proceedings of the 2010 conference on empirical methods in natural language processing, pp 218–228
24. Liu B, Dai Y, Li X, Lee WS, Yu PS (2003) Building text classifiers using positive and unlabeled examples. In: Proceedings of the third IEEE international conference on data mining (ICDM'03), pp 179–186
25. Liu B, Lee WS, Yu PS, Li X (2002) Partially supervised classification of text documents. In: Proceedings of the nineteenth international conference on machine learning, pp 387–394
26. Minsky M (1961) Steps toward artificial intelligence. Proc Inst Radio Eng 49:8–30
27. Pan S, Zhang Y, Li X (2012) Dynamic classifier ensemble for positive unlabeled text stream classification. Knowl Inform Syst 1–21. doi:10.1007/s10115-011-0469-2
28. Pazzani M (1997) Searching for dependencies in Bayesian classifiers. Springer, New York
29. Rodríguez JD, Pérez A, Lozano JA (2010) Sensitivity analysis of kappa-fold cross validation in prediction error estimation. IEEE Trans Pattern Anal Mach Intell 32(3):569–575
30. Sriphaew K, Takamura H, Okumura M (2009) Cool blog classification from positive and unlabeled examples. In: Proceedings of the 13th Pacific-Asia conference on advances in knowledge discovery and data mining, pp 62–73
31. Tsai RT-H, Hung H-C, Dai H-J, Lin Y-W, Hsu W-L (2008) Exploiting likely-positive and unlabeled data to improve the identification of protein-protein interaction articles. BMC Bioinform 9(Suppl 1):S3
32. Wang C, Ding C, Meraz RF, Holbrook SR (2006) PSoL: a positive sample only learning algorithm for finding non-coding RNA genes. Bioinformatics 22(21):2590–2596
33. Xiao Y, Segal MR (2008) Biological sequence classification utilizing positive and unlabeled data. Bioinformatics 24(9):1198–1205
34. Yousef M, Jung S, Showe LC, Showe MK (2008) Learning from positive examples when the negative class is undetermined-micro RNA gene identification. Algorithms Mol Biol 3:2
35. Yu H, Han J, Chang K (2004) PEBL: Web page classification without negative examples. IEEE Trans Knowl Data Eng 16(1):70–81
36. Yu H, Han J, Chang KC-C (2002) PEBL: positive example based learning for web page classification using SVM. In Proceedings of the Eighth ACM SIGKDD international conference on knowledge discovery and data mining, pp 239–248
37. Zhang D, Lee WS (2005) A simple probabilistic approach to learning from positive and unlabeled examples. In: Proceedings of Fifth annual UK conference on computational intelligence (UKCI), pp 83–87
38. Zhao X-M, Wang Y, Chen L, Aihara K (2008) Gene function prediction using labeled and unlabeled data. BMC Bioinform 9:57

Author Biographies

Borja Calvo received in 1999 his masters degree in Biochemistry from the University of the Basque Country (UPV/EHU). In 2004, he received his bachelor degree in Computer Science and in 2008 his PhD in Computer Science, both from the UPV/EHU. Currently, Borja Calvo holds a lecturer position at the Comptuer Science and Artificial Intelligence Department of the UPV/EHU. His research interests include machine learning and Bioinformatics.

Iñaki Inza received the PhD degree in computer science in 2002. Currently, he is a senior researcher enrolled in the Intelligent Systems Group, University of the Basque Country, San Sebastian, north of Spain. His main methodological research interests include classification and evolutionary computation by means of Bayesian networks, and feature selection. He has taken part in application projects in bioinformatics, web mining, and oceanographic domains.

Pedro Larrañaga received the diploma degree in mathematics from the University of Valladolid, Spain, in 1981, and the PhD degree in computer science from the University of the Basque Country, Spain, in 1995, where he became an associate professor in 1998 and full professor in 2004. In 2007, he joined the Technical University of Madrid as full professor in the Department of Artificial Intelligence, where he leads the Computational Intelligence Group. His research interests include probabilistic graphical models and heuristic optimisation. He has coauthored two edited books on estimation of distribution algorithms, as well as more than 300 scientific papers in different areas. He has participated in more than 70 research projects at national, European, and international levels.

Jose A. Lozano received the B.S. degree in mathematics and the B.S. degree in computer science from the University of the Basque Country, San Sebastian-Donostia, Spain, in 1991 and 1992, respectively, and the PhD degree in computer science from the University of the Basque Country in 1998. Since 2008, he has been a Full Professor with the University of the Basque Country, where he leads the Intelligent System Group. He is the co-author of more than 50 ISI journal publications and the coeditor of the first book published about estimation of distribution algorithms. His current research interests include machine learning, pattern analysis, evolutionary computation, data mining, metaheuristic algorithms, and real-world applications.