Using ensembles for problems with characterizable changes in data distribution: A case study on quantification

Pablo Pérez-Gállego, José Ramón Quevedo, Juan José del Coz

Artificial Intelligence Center, University of Oviedo, Gijón (Spain)

Abstract

Ensemble methods are widely applied to supervised learning tasks. Based on a simple strategy they often achieve good performance, especially when the single models comprising the ensemble are diverse. Diversity can be introduced into the ensemble by creating different training samples for each model. In that case, each model is trained with a data distribution that may be different from the original training set distribution. Following that idea, this paper analyzes the hypothesis that ensembles can be especially appropriate in problems that: (i) suffer from distribution changes, (ii) it is possible to characterize those changes beforehand. The idea consists in generate different training samples based on the expected distribution changes, and to train one model with each of them. As a case study, we shall focus on binary quantification problems, introducing ensembles versions for two well-know quantification algorithms. Experimental results show that these ensemble adaptations outperform the original counterpart algorithms, even when trivial aggregation rules are used.

Keywords: Distribution Changes, Ensembles, Quantification

1. Introduction

Ensemble learning consists in constructing a meta-model that results from the combination of a set of individual models, using a particular aggregation

Email address: `{pablopg,quevedo,juanjo}@aic.uniovi.es` (Pablo Pérez-Gállego, José Ramón Quevedo, Juan José del Coz)

rule. Ensembles get benefited from the existent diversity in the model set, producing a solution that implicitly represents some sort of agreement between the individual models. From a practical point of view, they generally perform better than a single-model solution [1, 2, 3, 4], although this cannot be guaranteed [5]. An ensemble limits the risk of obtaining a particular bad response from a single model; formally this is due to the fact that the ensemble tends to reduce the variance of its base classifier. Intuitively, the same idea is highly present in the human decision making processes; a set of opinions is more rich than an isolated opinion, especially when there exist a high degree of diversity within the opinions.

Let us introduce some notation for ensembles under the framework of supervised learning. Let $\mathcal{X}$ be an input space and $\mathcal{Y}$ an output space. There exist a training set $D = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_n, y_n)\}$ drawn from an unknown distribution $\mathbf{P}(\mathbf{X}, \mathbf{Y})$ from the product $\mathcal{X} \times \mathcal{Y}$. Usually each example, $\boldsymbol{x}_i$, is represented by an attribute vector $(x_{i,1}, x_{i,2}..., x_{i,d})$ and a target class y_i that may belong to a discrete set in classification problems or to $\mathbb{R}$ in the case of a regression problem. The objective is to approximate an unknown function $f : \mathcal{X} \to \mathcal{Y}$ by generating a function h, called hypothesis, defined into some hypothesis space $\mathcal{H}$. To do so, many algorithms search for the best single hypothesis h that approximates f taking into account the training set D, the selected hypothesis space and a target loss function. In contrast, an ensemble produces a hypothesis h resulting from the combination of a set of m (weak) hypothesis $\{h_1, h_2, ..., h_m\}$, in which each model h_j is usually learned using a subsample D_j generated from D. The combination of the set of hypothesis or models is performed by means of a particular aggregation strategy.

Supervised learning makes the assumption that the unknown distribution $\mathbf{P}(\mathbf{X}, \mathbf{Y})$, from where the examples are drawn independently and identically distributed (i.i.d. assumption), does not change between the training and testing or production phases. Or state it differently, it is assumed that the training set truly reflects the probability distribution of the problem. However, in practice, this assumption gets often violated in real-world applications [6, 7]. This situation is referred to as *dataset shift* in the research community, and it takes place when $\mathbf{P}(\mathbf{X}, \mathbf{Y})$ changes from training to testing data [8, 9]. Characterizing those changes in the distribution sometimes depend on the target application, and even though it can be challenging in some cases, it is surprisingly simple, even trivial, in other problems.

Our intention in this paper is to present a new scenario in which the application of ensemble algorithms results appropriate and effective. We refer to problems verifying the next properties: (i) are known to suffer from distribution changes between training and testing phases, (ii) we are able to characterize the distributional changes beforehand (i.e. we can define the conditions that make $\mathbf{P}(\mathbf{X},\mathbf{Y})$ to change). The objective is to take advantage of this a priori knowledge and to use it during the training stage. From an ensemble point of view, we can make the most of this knowledge in order to introduce *diversity* into the weak models, a desirable property to boost its efficacy [10, 11, 12, 13, 14]. The central idea of this paper is to generate different training samples, with each one representing an specific and expected distribution change. This approach is different to other propositions that have been suggested to tackle tasks that present some sort of drift in the distribution [15], especially *concept drift* problems [16]. Those methods are based on removing, modifying or adding new models to the ensemble, mainly because the concept $\mathbf{P}(y|\boldsymbol{x})$ changes throughout time and it is not possible to exploit any prior knowledge. Our approach is different in the sense that, as we know the characteristics of the expected changes, we can use that knowledge to build an enriched ensemble from the beginning without the need of subsequent modifications.

In order to prove the validity of our idea, we have applied it to *binary quantification* problems. Quantification is defined as the task of estimating the number of examples belonging to each class (class distribution) in a test set, using a training set that may have been drawn from a different distribution [17]. In the case of binary quantification the set of class values is restricted to two and the objective is to correctly estimate the number of positive examples (prevalence). Quantification tasks fit perfectly our requirements, since, by definition, the class probabilities may change, and we are also able to characterize and to restrict ourselves to a certain types of changes, as we shall see later.

There are many real-world problems that can be solved using quantification algorithms. Tentative application scopes include opinion mining [18], network-behavior analysis [19], quality control [20], monitoring of support-call logs [21] and credit scoring [22], among others. For instance, there is an increasing demand for automatic methods to track overall consumer opinions [23]. The goal is to answer questions like *how many consumers are satisfied with our new product?*. This task requieres effective algorithms focused on estimating the distribution of classes from a sample. Notice that the goal is not to label individual examples (solved using traditional classification algorithms), but to obtain estimations at aggregated level. This kind of problems are related with those aimed at tracking trends over time [24], such as early detection of epidemics and endangered species, risk prevalence and ecosystems evolution.

In the experimental results section we empirically demonstrate that the estimates provided by a single quantifier can be improved by using its ensemble version. We have compared the performance of our ensemble quantifier approach with a baseline quantifier, *CC* (*Classify and Count*)[25], and two state-of-the-art quantifiers, *AC* (*Adjusted Count*)[25] and *HDy*[26]. Nevertheless, we think that the significance of this article goes beyond that fact, since the proposed approach can be applied to different distribution change problems, as long as it is possible to characterize those changes beforehand. Interestingly, several studies characterizing some of these problems have been published recently [27, 8, 9, 28].

The rest of this paper is organized as follows: Section 2 briefly describes distribution changes and how to characterize them and Section 3 introduces the binary quantification problem and the quantification algorithms used in this paper. In Section 4 the details of our ensemble quantification approach are presented. The experimental setup and empirical results are shown in Section 5. Section 6 summarizes the main conclusions.

2. Characterizing problems with changes in data distribution

A categorization and a discussion about problems presenting distribution changes, or using the current terminology, problems suffering from *dataset shift*, can be found for instance in [8, 9]. Supervised learning problems are defined by a set of covariates, $\boldsymbol{x}$, a class variable, y, and the examples are drawn at random from the joint probability distribution of both. To better understand dataset shift it is important to realize how the data is generated according to the causal relationship between covariates and the class variable, since it determines the kind of changes in the distribution that a problem may suffer from. In this sense, a taxonomy proposed in [29] identifies two types of problems: $\mathcal{X} \rightarrow \mathcal{Y}$ in which the class value y is causally determined by the covariate values $\boldsymbol{x}$, and problems $\mathcal{Y} \rightarrow \mathcal{X}$ where the covariates $\boldsymbol{x}$ causally depend on the class label y. Spam detection constitutes an example of the first type of problems, the mail content determines whether it is spam or not. On the other hand, medical diagnosis problems are a typical example of the second; suffering from a determined disease y cause a series of symptoms $\boldsymbol{x}$ to appear.

Given an instance $\boldsymbol{x}$ and a class value y, their joint probability $\mathbf{P}(\boldsymbol{x}, y)$ can be written as $\mathbf{P}(y|\boldsymbol{x})\mathbf{P}(\boldsymbol{x})$ in $\mathcal{X} \rightarrow \mathcal{Y}$ problems and $\mathbf{P}(\boldsymbol{x}|y)\mathbf{P}(y)$ in the case of $\mathcal{Y} \rightarrow \mathcal{X}$ problems. Dataset shift arises when any of these elements change between training and test, that is to say $\mathbf{P}_{tr}(\boldsymbol{x}, y) \neq \mathbf{P}_{tst}(\boldsymbol{x}, y)$. Thus, several types of dataset shift problems can be identified depending on the elements that change:

- *covariate shift*: $\mathbf{P}(\boldsymbol{x})$ changes but $\mathbf{P}(y|\boldsymbol{x})$ remains constant
- *prior probability shift*: $\mathbf{P}(y)$ changes but $\mathbf{P}(\boldsymbol{x}|y)$ does not
- *concept shift* (o *drift*): $\mathbf{P}(y|\boldsymbol{x})$ changes but $\mathbf{P}(\boldsymbol{x})$ does not ($\mathcal{X} \rightarrow \mathcal{Y}$ problems), or $\mathbf{P}(\boldsymbol{x}|y)$ changes but $\mathbf{P}(y)$ remains constant ($\mathcal{Y} \rightarrow \mathcal{X}$ problems)

Supervised learning methods generally assume that the joint probability distribution remains unaltered between training and test. However, in practice, there are many important applications suffering from changes to a greater or lesser extent. These kind of problems are interesting from an ensemble learning point of view because some of the aforementioned changes can be easily characterized. This is especially true in the case of *prior probability shift* problems, that are also referred to as *quantification* problems in the literature. A typical application of quantification learning is to estimate the prevalence of positive and negative opinions. Imagine that we want to track the opinions about a product in Twitter during a period of time and give just an estimate on how many are positives (and negatives), without predicting individual opinions (this would be a classification task). In such a problem, when the class distribution $\mathbf{P}(y)$ changes (e.g. the number of positive opinions increases), the opinions maintain the same distribution when the class is fixed. The way in which users express their opinions does not change from one day to another, there would be very good opinions using strong words expressing that felling, moderately positive comments and so on. When can assume in that case that $\mathbf{P}(\boldsymbol{x}|y)$ is constant.

As we state in the previous section, ensembles have been applied before for problems that present a shift in the distribution, mainly in concept drift tasks [30]. The main idea is to build an ensemble with models created at different moments in time and the goal is to have at least one model representing each distinct concept. The ensemble maintains a *memory* of models representing past concepts because some of them may become useful again in the future

[31]. Different strategies for training such ensembles can be employed, for instance, to divide historical sequential data into non overlapping blocks [32, 33, 34] or using different sized training windows [35, 36, 37]. After the set of models is trained, adaptivity is achieved by defining a combination or fusion rule. Basically, the combination rule consists in assigning weights to the individual models at each point in time. The weights express the expected competence of the model and may depend on different factors, typically the estimated performance [33, 37] or historical performance in the past [34, 35, 36].

However, our approach differs in several aspects with respect to these concept drift methods. The first difference is due to the fact that the concept does not change in quantification applications. For instance, the concept of what a positive opinion is does not change for a given sentiment analysis problem. The ensembles in concept drift tasks are usually designed to maintain a memory of models, representing the evolution of the concept and allowing to reuse models that were obtained in the past and are valid again. This approach is not aplicable for quantification tasks because because the concept is always the same, there are no *old* and *new* models in quantification. This is particularly true for $\mathcal{Y} \to \mathcal{X}$ problems for obvious reasons. The second difference is that models for concept drift based on ensembles are trained with successive samples. In our case, the samples are not given but are generated according to the expected changes in data distribution.

3. Binary quantification

Given a training set with examples labeled as positives or negatives, $y_i \in \{+1, -1\}$, the class distribution can be summarized with the actual proportion of positives or *prevalence* p (the proportion of negatives is $n = 1-p$). The binary quantification goal is to induce a model or quantifier able to give an estimate p' of the actual prevalence, $p = \mathbf{P}(y = +1)$, for a test set T that may have a significantly different class distribution (see Figure 1). From a learning point of view, the most important assumption made by quantification methods is that the class probability distribution $\mathbf{P}(y)$ changes between training and test, but $\mathbf{P}(\boldsymbol{x}|y)$ remains constant.

Intuitively, it seems a less complex problem than classification, since it is not necessary to give accurate estimations for each example. We could think of a first trivial approach to quantification by inducing a classifier, predicting the class of each example, and counting the number of predicted

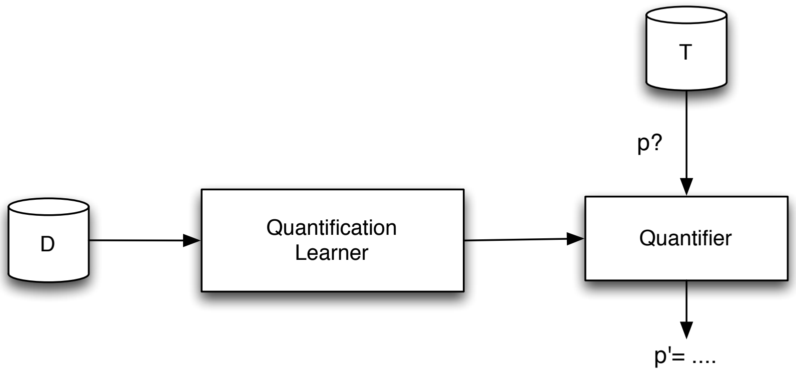


Figure 1: Quantification learning

examples for each class, obtaining a class histogram. This method is referred to as *CC* (*Classify* and *Count*) and has already been proved to perform poorly [17]. The problem of such approach is that the induced classifier does not take into account that the distribution may change, which, by problem definition, will happen. In fact, *CC* performance tends to notably drop when p significantly changes, by going down or up a high degree. The main reason is because the underlying classifier will have a systematic bias, tending to produce false positives or false negatives. For instance, the derived quantifier will underestimate the prevalence, $p' \ll p$, when p increases and the classifier tends to produce false negatives. And the opposite, the quantifier will also overestimate p, $p' \gg p$, when p decreases and the classifier produces more false positives. Thus, it is impossible for *CC* to perform well in the whole prevalence domain.

In this paper we have chosen to work with two quantification methods following contrasting approaches.

3.1. Adjusted Count

The first one is named *AC* (*Adjusted Count*) and was proposed by Forman [17] in order to reduce the systematic bias of *CC* inherited from the underlying classifier. Noticeably, the same idea was presented back in the 70's for the estimation of class prevalence from screening tests in epidemiology [38]. The differences with respect to Forman's proposal are two: i) no supervised learning is involved and (ii) the role of the classifier is played by a clinical test that has imperfect sensitivity and specificity.

AC is based on estimating the classifier *tpr* (*true positive rate*) and *fpr* (*false positive rate*):

$$tpr = \frac{TP}{P}\ ,\ fpr = \frac{FP}{N} \tag{1}$$

in which P and N represent the count of actual positives and negatives, while TP and FP represent the count of true positives and false positives predicted by the model. These two values can be used to correct the prevalence estimation given by a classifier. In fact, the probability of a classifier making a positive prediction, $p' = \mathbf{P}(h(\boldsymbol{x}) = +1)$, in a binary problem can be expressed as a function of the ground truth prevalence p:

$$\begin{aligned} p' &= \mathbf{P}(h(\boldsymbol{x})=+1|y=+1)\cdot \mathbf{P}(y=+1) + \mathbf{P}(h(\boldsymbol{x})=+1|y=-1)\cdot \mathbf{P}(y=-1) \\ &= tpr\ \cdot\ p + fpr\ \cdot\ n \\ &= tpr\ \cdot\ p + fpr\ \cdot\ (1-p) \\ &= fpr + p\ \cdot\ (tpr - fpr). \end{aligned} \tag{2}$$

Solving for p, the ground truth prevalence can be written depending on p', *tpr* and *fpr*:

$$p = \frac{p' - fpr}{tpr - fpr}. \tag{3}$$

Therefore, in order to obtain a better estimation of the true prevalence p it will suffice to (i) train a classifier, (ii) estimate its *tpr* and *fpr*, (iii) classify and count the test examples towards obtaining p' and (iv) obtain the true prevalence p by substituting the calculated values in (3). The crucial step, which is the estimation of *tpr* and *fpr*, is usually performed using cross-validation over the training set [17]. It is important to emphasize that such estimations are independent of the distribution changes, since it was assumed that $\mathbf{P}(\boldsymbol{x}|y)$ remained constant. Notice that theoretically AC should output perfect quantification estimates as long as this assumption is fullfiled. However, in practice, inaccurate *tpr* and *fpr* estimations and/or the assumption not getting satisfied often lead to imperfect quantifications.

The AC correction is, in general, applicable to any quantifier built on top of a classifier, like in the case of quantifiers based on decision trees [39], on nearest neighbors [40], on structured classifiers that optimize quantification measures [41] or even on probabilistic classifiers, with the necessity of adapting the correction to the probabilistic domain [42].

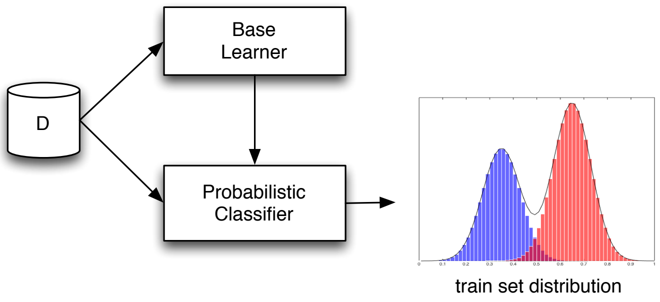


Figure 2: *HDy* Training phase: a probabilistic classifier is learned and applied to obtain the distribution of D

3.2. HDy

The second method, termed *HDy* [26], is totally different to the previously commented approach, since it does not rely on classifying, counting and correcting. Its core idea consists in measuring distribution similarities. More precisely, *HDy* compares training and test distributions and seeks for the prevalence p' that would make the training distribution, modified by p' and obeying that $\mathbf{P}(\boldsymbol{x}|y)$ does not change, the most similar to the test distribution.

In [26], the authors propose two different methods to represent both distributions: *HDx* uses directly the feature vectors $\boldsymbol{x}$, while *HDy* employs the predictions of a probabilistic model induced from the training set. The later not only outperforms the former, but it also has the advantage of working in a space with a smaller dimensionality. In fact, the main advantage of *HDx* is that it does not require a classification model, but its main limitation is that its computational complexity increases with the number of features. Thus, the use of *HDx* is unfeasible in high dimensional spaces.

In the case of *HDy* its representational space has just $M - 1$ dimensions, being M the number of classes. For a binary problem it has only one dimension: the model predicts the probability of an example to belong to the positive class. The process is quite simple: the range $[0..1]$ is partitioned into b bins, and a histogram is built with each classified example being assigned to one of the bins depending on its probabilistic score. In the training stage, the probabilistic classifier and the actual distribution of the training set are obtained (see Figure 2).

In the testing phase, the distribution of the test set is obtained following the same procedure: (i) computing the probabilistic scores of the testing examples using the probabilistic classifier, and (ii) calculating the corresponding histogram using the same number of bins b. Then, the idea is to modify the training histogram to match the testing histogram varying p'. For instance, in the example depicted in Figure 3 in which red color represents the distribution of the positives, p' should decrease (and of course $n' = 1 - p'$ should increase in the same quantity) with respect to the prevalence of the training set in order to match the testing distribution.

The strategy followed for modifying the original training set histogram is a simple linear search. p' is moved over the range [0..1] in small steps and the histogram associated is calculated applying the following equation:

$$\frac{|D'_i|}{|D'|} = \frac{\left|D_i^+\right|}{|D^+|} \cdot p' + \frac{\left|D_i^-\right|}{|D^-|} \cdot (1 - p'), \tag{4}$$

in which $|D^+|$ is the number of examples in D belonging to the positive class and $\left|D_i^+\right|$ the number of positive examples in D belonging to the i-th bin. $|D^-|$ and $\left|D_i^-\right|$ are analogously referred to the negative class. These components are pre-calculated in the training phase (Figure 2).

Similarity between both histograms is measured with the *Hellinger Distance* metric:

$$HD(D', T) = \sqrt{\sum_{i=1}^{b} \left(\sqrt{\frac{|D'_i|}{|D'|}} - \sqrt{\frac{|T_i|}{|T|}} \right)^2}, \tag{5}$$

where D' and T represent the modified distribution of the training set computed using (4) and the testing distribution, respectively. $|D'_i|$ is the number of examples of the modified distribution D' belonging to the i-th bin and $|D'|$ is the training set cardinality. $|T_i|$ and $|T|$ refer in the same way to the test set distribution.

Figure 3 schematically shows the testing phase of *HDy*. For the different values of $p' \in [0..1]$ considered, the Hellinger distance to the test set is calculated using (5). The p' value minimizing the distance to the test set will be the estimated prevalence. Note that, as all the bins of the train set distribution are altered uniformly using p', the assumption of $\mathbf{P}(\boldsymbol{x}|y)$ being constant is not violated.

4. Ensembles for problems with characteristic distribution changes

As we have pointed out in previous sections, there exist problems, like covariate shift or quantification, in which we: (i) are aware of the distribution

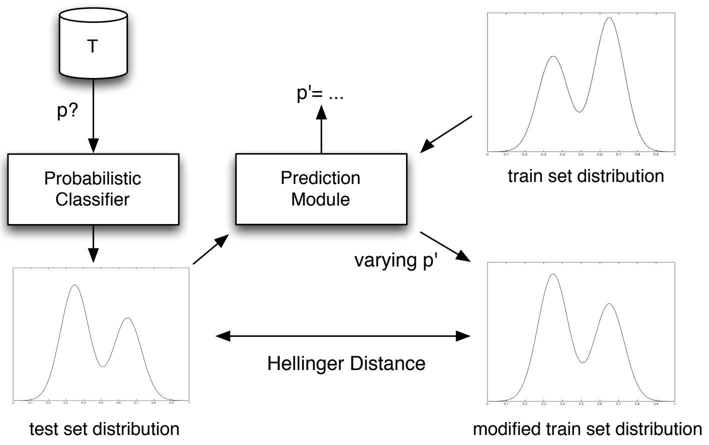


Figure 3: *HDy* Testing phase: the test distribution is computed using again the probabilistic classifier. The most important step is to compute a modified distribution of D, varying p' in (4). Similarity between both distributions is calculated using the Hellinger Distance (5)

changing between training and test, (ii) are able to characterize the kind of changes to emerge, or at least (iii) can make a previous assumption about the nature of such changes. Thus, the core idea in this paper is that we can take advantage of that knowledge to build ensembles with an appropriate diversity that are prepared to effectively represent the expected changes in the distribution. The ensemble learning task is separated into three phases: (i) sample generation with each one representing an expected distribution change, (ii) model training on each generated sample and (iii) combination of the individual estimates to produce the final ensemble prediction.

The most important step is certainly the first one, which must be adapted depending on the dataset shift problem being tackled. In this article we are showing a simple adaptation for the binary quantification problem. In other cases the adaptation may require from additional knowledge about the particular application nature, as in the case of covariate shift, with $\mathbf{P}(\boldsymbol{x})$ being the changing component. In the binary quantification case it is more straightforward. It is important to keep in mind the learning assumption of the problem: quantification mainly stands for $\mathcal{Y} \rightarrow \mathcal{X}$ problems with $\mathbf{P}(y)$ changing and $\mathbf{P}(\boldsymbol{x}|y)$ remaining constant. Once the expected distribution changes are defined, the goal is to generate training samples a priori representing them, so it is possible to effectively react to its presence. Thus, sampling is not aimed

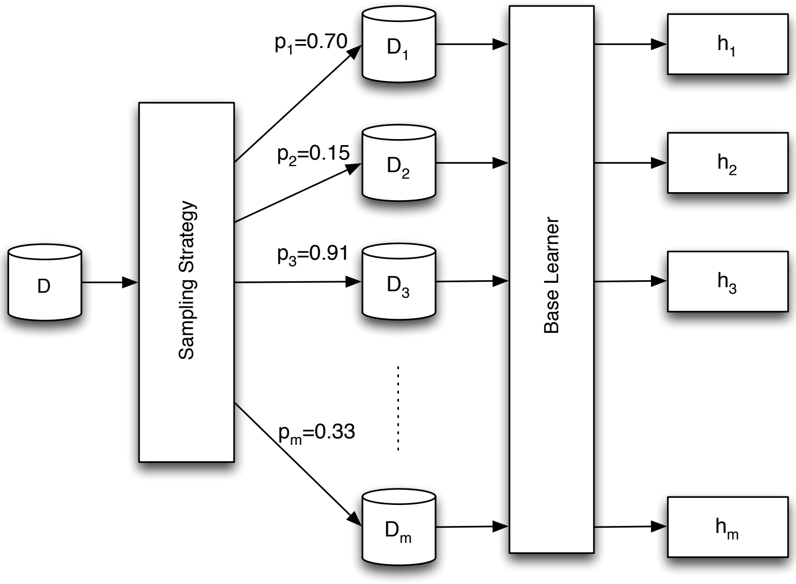


Figure 4: Training stage of the ensemble. Each sample D_j is generated with a different prevalence p_j

at correctly predicting concrete examples, even though it is a characteristic to also be considered in the future, but just at representing the expected distribution changes.

We propose the following procedure in order to generate each training sample. First, the sample prevalence p_j is randomly selected in [0..1]. Then, simple random sampling with replacement is performed within the positive class examples until the number of positive examples, given by the chosen prevalence p_j, is obtained. This process guarantees $\mathbf{P}(\boldsymbol{x}|y)$ to be constant. The same operation is repeated for the negative class, with its prevalence being equal to $n_j = 1 - p_j$. By changing the prevalence of each generated sample we obtain the desired diversity. This procedure is repeated until the number of defined training samples m is reached. Figure 4 describes this process.

The next step is to train the base quantifier algorithm over each generated sample. In the case of the quantifier using the AC correction (3) it is necessary to estimate the *tpr* and the *fpr* using each sample D_j. An important detail to highlight is that the generated samples D_j should have an adequate

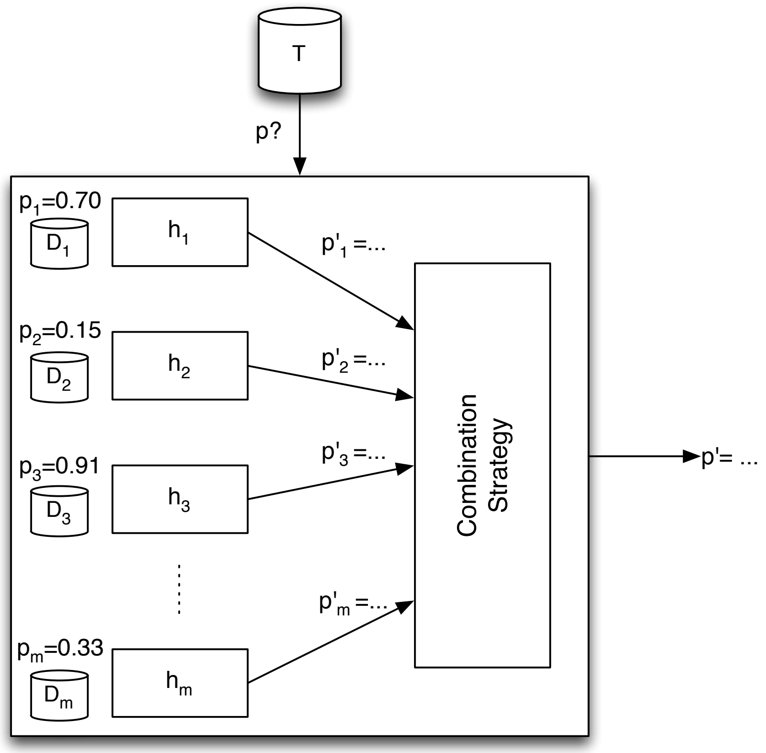


Figure 5: Prediction stage of the proposed ensemble for binary quantification. Each model h_j is applied over test set T, obtained different values for the estimated prevalence (p'_j) of T. These values are combined using a combination strategy to obtain the final estimate p'

number of examples in order to be able to produce accurate estimates for *tpr* and *fpr*. In our case, we generate samples with the same cardinality as the original training set but with a different prevalence in general.

Finally, given an unlabeled test set T, each ensemble learned model h_j is applied and an estimate of the test set prevalence, p'_j, is obtained. The definitive ensemble prediction is computed by applying a combination function. In the experiments reported below we have just used a simple aggregation function, the arithmetic mean. Nevertheless we think that more complex and powerful methods can be investigated in the future.

We believe that applying ensemble techniques to the quantification problem results in more benefits than its application in other kind of problems, like classification. The advantage of using a combination of models instead of a single model, with the risk of not performing well in some cases, is generic to every kind of problem in which ensemble technique is applied to. However, in the quantification case there are additional benefits. First of all, we actually know how to explicitly introduce diversity into the ensemble. In the second place, using multiple models when the base quantifier corrects its estimates leads to major improvements. The AC correction (3), while theoretically produces perfect quantifications, is, in practice, risky. When having inaccurate *tpr* and *fpr* estimates, the correction will cause inappropriate changes. As we shall show in the experimental results section, AC manages to produce very accurate results for some problems, while it significantly drops its performance for others. It not only depends on the underlying classifier accuracy, but principally on the correction quality. Using ensembles reduces both risks. Not only the diversity itself results in an ensemble specific benefit, but also the combining strategy lessens more risks than in a typical classification scenario.

5. Experiments

The goal of the performed experiments was to empirically verify the effectiveness of applying ensembles to a problem noticeably suffering from data distribution changes, as it is the case of quantification. These experiments compare the performance of a baseline quantifier, *CC*, and two state-of-the-art quantification algorithms, AC and *HDy*, to the performance of its ensemble adapted versions, *ECC*, *EAC* and *EHDy*. The aim is to improve the performance of AC and *HDy* using ensembles.

5.1. Experimental design

We basically chose AC and *HDy* as quantifier representatives because they are based on very contrasting approaches, thus increasing the study significance. On the one hand, AC, as an emblematic quantification algorithm based on classifying and counting with posterior estimate corrections using *tpr* and *fpr*. On the other hand, *HDy*, even though it also makes use of a classifier to represent sample distributions, it is based on measuring distribution similarities rather than in classifying, counting and correcting. Additionally, other three ensemble-based algorithms, named as *CC(bag)*, *AC(bag)* and *EHDy(bag)*, were included in the experiments to better study the influence of the design decisions made to propose *EAC* and *EHDy*. These three methods use bagging [2] as the underlying classifier, applied with different base learners.

The algorithms compared can be grouped according to two different criteria. The main criterion depends on the quantification approach: *CC* methods (*CC*, *CC(bag)* and *ECC*), *AC* methods (*AC*, *AC(bag)* and *EAC*) and *HDy* methods (*HDy*, *HDy(bag)* and *EHDy*). Notice that the difference between the algorithms in the same group are due to the different classifier or approach used. But from another point of view, we can group the algorithms considering the models they use. This is important for the fairness of the experiments. First, we have those quantifiers that only need a single non-ensemble classifier. To this group belongs *CC*, *AC* and *HDy*. In the experiments all these methods share the same underlying model in order to perform a fair comparison between them. The second group is formed by *CC(bag)*, *AC(bag)* and *EHDy(bag)*. The bagging classifier is also the same. Finally, *ECC*, *EAC* and *EHDy* implement the ensemble proposed in the paper, using samples with a different prevalence. Again, the ensemble is composed by exactly the same models. Hence, the differences in their performance between the algorithms in the same group according to the second criterion are only due to the different way in which the predictions made by the classifier are used; the differences between them are not due to the classifier.

The experiments were carried out over 32 datasets, specifically all of those used in [40, 26], except iris.1, acute.a and acute.b (perfect quantifiers are obtained, so they are trivial problems), and coil, lettersG and lettersH because the experiments did not finish in a reasonable time. As we shall see in this description of the experimental setting used, the experiments reported are quite complex due to several reasons, mainly the use of ensembles, the grid-search applied for tuning classifiers' parameters and the testing phase, which is more time consuming than in classification problems because we need to estimate the prevalence for different test sets. The testing phase is even slower when using ensembles. Table 1 contains the characteristics of the datasets finally used. Some of them are originally binary, while others are binary adaptations of multiclass problems. For instance, iris.2 is a binary problem in which the original class 2 has been mapped to the positive class, with the remaining classes being mapped to the negative class. In all cases, the instances with missing values were removed.

We decided to use probabilistic base classifiers because *HDy* uses the scores provided by the underlying classifier to represent the distributions. This decision avoids unbounded predictions and homogenizes the histograms

Table 1: Summary of datasets

Dataset	*Identifier*	*Size*	d	P	N	p
Balance Scale (left)	balance.1	625	4	288	337	46%
Balance Scale (balanced)	balance.2	625	4	49	576	8%
Balance Scale (right)	balance.3	625	4	288	337	46%
Breast Cancer Wisconsin	breast-cancer	683	9	444	239	65%
Contraceptive Method Choice (no use)	cmc.1	1473	9	629	844	43%
Contraceptive Method Choice (long term)	cmc.2	1473	9	333	1140	23%
Contraceptive Method Choice (short term)	cmc.3	1473	9	511	962	35%
Cardiotocography Data Set (normal)	ctg.1	2126	22	1655	471	78%
Cardiotocography Data Set (suspect)	ctg.2	2126	22	295	1831	14%
Cardiotocography Data Set (pathologic)	ctg.3	2126	22	176	1950	8%
Pima Indians Diabetes Data Set	diabetes	768	8	268	500	35%
Statlog German Credit Data	german	1000	24	700	300	70%
Haberman's Survival Data	haberman	306	3	81	225	26%
Johns Hopkins University Ionosphere DB	ionosphere	351	34	126	225	36%
Iris Plants Database (versicolour)	iris.2	150	4	50	100	33%
Iris Plants Database (virginica)	iris.3	150	4	50	100	33%
Mammographic Mass	mammographic	830	5	403	427	49%
Page Blocks Classification (5)	pageblocks.5	5473	10	115	5358	2%
Phoneme	phoneme	5404	5	1586	3818	29%
Semeion Handwritten Digit (8)	semeion	1593	256	155	1438	10%
Sonar, Mines vs. Rocks	sonar	208	60	97	111	47%
Spambase Data Set	spambase	4601	57	1813	2788	39%
SPECTF Heart Data	spectf	267	44	55	212	21%
Tic-Tac-Toe Endgame Database	tictactoe	958	9	332	626	35%
Blood Transfusion Service Center Data Set	transfusion	748	4	178	570	24%
Wisconsin Diagnostic Breast Cancer	wdbc	569	30	212	357	37%
Wine Recognition Data (1)	wine.1	178	13	59	119	33%
Wine Recognition Data (2)	wine.2	178	13	71	107	40%
Wine Recognition Data (3)	wine.3	178	13	48	130	27%
Wine Quality Red (6-10)	wine-q-red	1599	11	855	744	53%
Wine Quality White (6-10)	wine-q-white	4898	11	3258	1640	67%
Yeast	yeast	1484	8	429	1055	29%

obtained. Such histograms were generated with 8 bins. The results of *HDy* with other values for this parameter were similar (with 4 bins), or worse (with 12 and 20 bins).

Three different base classifiers were employed to verify that the results do not depend on a particular classifier. Each base learner is applied in combination with the nine methods compared. The learners selected were: (i) Naïve Bayes (*NB*), the most classical probabilistic classifier, (ii) Logistic Regression (*LR*), to induce linear models and (iii) *SVM* with a Gaussian kernel (*RBF*) and probabilistic output to produce non linear models. In the case of *LR* and *SVM-RBF* the regularization parameter (C) was set through a search in $C \in [10^{-3}, \ldots, 10^{3}]$, by optimizing the geometric mean in binary classification using a 5-fold cross validation with 2 repetitions (CV5x2). As regards *SVM-RBF*, the parameter γ was selected within the values $[0.001, 0.005, 0.01, 0.05, 0.1, 1]$ following the same procedure. *NB* uses gaussian distributions to deal with numerical attributes.

Geometric mean measures the ability of a classifier to balance sensitivity (accuracy on positive examples) and specificity (accuracy on the negative examples). It was selected to deal with imbalance datasets, see details in Table 1. Moreover, in the case of *LR* and *SVM-RBF* we equally weighted both classes by using the $-w$ parameter in LibLinear and LibSVM [43]. The weight of each class multiplies the regularization parameter C, so both classes have the same influence in the loss term of the optimization problem. Both decisions are tailored at obtaining well suited classifiers even under important class imbalance scenarios, a usual situation in quantification problems. Without class weighting, and by only optimizing the classifier accuracy, both CC and AC performed significantly worse and the difference with respect to the ensemble versions gets accentuated, due to the classifier tending to predict the majority class in most of the cases.

The estimation of *tpr* and fpr, a critical step for AC-based approaches, was obtained through a 10-fold cross-validation over the training examples. Notice that for EAC, each model of the ensemble requires to estimate its *tpr* and fpr. The size of the ensemble approaches was set to $m = 30$. We decided to generate samples D_j with the same size as the original training set D in order to be able to get accurate estimates of the *tpr* and the *fpr* when needed. The prevalence p_i was randomly chosen in the interval $[5\% - 95\%]$ for each sample. We deliberately avoided values near 0% and 100% because in these points it may arise difficulties when estimating both *tpr* and *fpr*, due to the lack of examples in one of the classes. Given the selected prevalence p_i, random sampling with replacement (to ensure $\mathbf{P}(\boldsymbol{x}|y)$ remains constant) was used to generate the number of examples required for each class.

The results reported in the next section were obtained using 5-fold cross validation with two repetitions (CV5x2). For each test partition 100 samples were generated with replacement, in which the positive class prevalence p was randomly selected ranging from 0% to 100%. Thus, each result in the tables represents the mean of 1000 quantifications. The error metric represented is the mean squared error (MSE). Similar results are obtained by analyzing the mean absolute error (MAE).

Table 2: Mean squared error using Logistic Regression as base classifier. The score of the best performer in a group for each dataset is in bold

dataset	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
balance.1	0.0031	0.0031	**0.0025**	0.0025	0.0025	**0.0021**	0.0014	0.0013	**0.0012**
balance.2	0.1309	0.1436	**0.1123**	0.1833	0.2185	**0.1328**	0.3708	0.3298	**0.1530**
balance.3	0.0021	0.0021	**0.0019**	**0.0010**	**0.0010**	**0.0010**	**0.0006**	**0.0006**	**0.0006**
breast-cancer	**0.0008**	0.0010	0.0009	0.0007	0.0008	**0.0006**	**0.0004**	**0.0004**	**0.0004**
cmc.1	**0.0436**	0.0439	0.0451	0.0123	0.0121	**0.0114**	0.0118	0.0117	**0.0105**
cmc.2	**0.0429**	**0.0429**	0.0432	0.0164	0.0167	**0.0148**	0.0109	0.0107	**0.0100**
cmc.3	**0.0521**	0.0524	0.0538	0.0318	0.0281	**0.0238**	0.0184	0.0178	**0.0156**
ctg.1	**0.0056**	**0.0056**	0.0059	0.0016	0.0017	**0.0013**	0.0006	**0.0005**	0.0007
ctg.2	0.0092	**0.0080**	0.0110	0.0013	**0.0010**	**0.0010**	**0.0010**	**0.0010**	0.0011
ctg.3	**0.0020**	0.0025	0.0024	**0.0011**	0.0014	0.0013	0.0015	0.0017	**0.0014**
diabetes	0.0211	**0.0208**	0.0229	0.0102	0.0071	**0.0066**	0.0057	**0.0051**	**0.0051**
german	**0.0274**	**0.0274**	0.0300	0.0091	0.0094	**0.0085**	**0.0070**	0.0071	0.0075
haberman	0.0777	0.0757	**0.0715**	0.0780	0.0773	**0.0662**	0.0966	0.0856	**0.0735**
ionosphere	**0.0152**	0.0236	0.0213	**0.0068**	0.0088	0.0108	**0.0131**	0.0148	0.0185
iris.2	0.0448	0.0492	**0.0438**	0.0677	0.0765	**0.0457**	0.0329	0.0326	**0.0189**
iris.3	0.0018	**0.0014**	0.0015	0.0017	**0.0009**	**0.0009**	0.0017	0.0014	**0.0011**
mammographic	0.0168	0.0163	**0.0148**	0.0101	0.0077	**0.0068**	0.0048	**0.0042**	0.0044
pageblocks.5	0.0078	**0.0077**	0.0264	0.0046	**0.0043**	0.0052	0.0016	0.0016	**0.0010**
phoneme	**0.0253**	0.0255	0.0254	**0.0017**	0.0019	0.0018	0.0010	**0.0009**	0.0010
semeion.8	**0.0070**	0.0089	0.0082	**0.0017**	0.0024	0.0046	0.0054	0.0042	**0.0037**
sonar	**0.0230**	0.0250	0.0254	0.0357	0.0387	**0.0187**	0.0362	0.0333	**0.0220**
spambase	0.0079	0.0050	**0.0047**	0.0021	0.0004	**0.0003**	**0.0002**	**0.0002**	**0.0002**
spectf	**0.0393**	0.0407	0.0408	0.0424	0.0515	**0.0278**	0.0549	0.0475	**0.0332**
tictactoe	0.0480	**0.0476**	0.0492	0.0365	0.0369	**0.0289**	0.0203	0.0198	**0.0172**
transfusion	0.0471	**0.0441**	0.0446	0.0293	0.0239	**0.0194**	**0.0214**	0.0234	0.0230
wdbc	0.0066	0.0066	**0.0063**	0.0037	0.0033	**0.0031**	**0.0023**	0.0025	0.0024
wine.1	0.0042	0.0044	**0.0037**	0.0035	0.0038	**0.0026**	0.0024	0.0027	**0.0021**
wine.2	0.0094	0.0067	**0.0052**	0.0089	0.0057	**0.0041**	0.0058	0.0041	**0.0038**
wine.3	0.0016	0.0016	**0.0014**	0.0021	0.0020	**0.0015**	0.0009	0.0010	**0.0008**
wine-quality-red	0.0234	**0.0232**	0.0238	0.0082	**0.0062**	0.0069	0.0051	**0.0044**	0.0047
wine-quality-white	0.0333	**0.0326**	0.0328	0.0030	0.0027	**0.0023**	0.0017	0.0016	**0.0015**
yeast	**0.0308**	0.0316	0.0321	0.0088	0.0085	**0.0072**	0.0043	0.0045	**0.0039**
Average	0.0254	0.0260	0.0255	0.0196	0.0207	0.0147	0.0232	0.0212	0.0139
Average ranking	7.1875	7.2500	7.0625	5.5469	5.3125	3.5156	3.7500	3.1719	2.2031

5.2. Experimental results

Table 2 shows the scores for each method using Logistic Regression as base classifier. It is worth noting that *EAC* and *EHDy* obtains the best results overall, not only better than those of their counterparts, *AC* and *HDy*, but also better than the scores of the bagging methods (*AC(bag)* and *HDy(bag)*). Analyzing the results from that point of view, it seems that the use of ensembles helps to produce a slight improvement in performance, comparing *AC* vs. *AC(bag)* and *HDy* vs. *HDy(bag)*. This improvement is boosted when the proposals of this paper are applied, as we can see analyzing the differences between *AC(bag)* and *EAC* and between *HDy(bag)* vs. *EHDy*.

If we focus on the comparison between the original quantifiers, *AC*/*HDy*, and the ensemble adapted versions, *EAC*/*EHDy*, we can observe the ensemble versions winning most of the times. With respect to the *EAC* vs *AC* comparison, the former wins on 26 occasions, losses just 5 times and there is only one draw. As concerns *EHDy*, it wins on 22 occasions to *HDy*, losses on 6 and there are 4 draws. A more detailed examination reveals that ensemble versions defeats are usually produced by small margins in low quadratic error problems, while, in contrast, in more difficult problems generally they get the victory by relatively large margins. These results are relevant since *AC* is supposed to produce, at least theoretically, perfect quantifications independently of the classifier accuracy. However, in practice, the correction (3) often works well in a range close to the training set prevalence, but its effectiveness drops as the test set prevalence moves away. Using ensembles reduces the impact of this situation, since the training sample generation process ensures all the prevalence domain to be represented within the ensemble models.

Finally, the trivial approximation of classifying and counting, provided by (*CC*)-based methods, attains worse scores than the other approaches. This result is in line with the experiments reported in quantification literature, in which *CC* is always outperformed by proper quantifiers. Notice that the performance of *CC* does not improve even when an ensemble is used as classifier, in fact, the scores of the three methods (*CC*, *CC(bag)* and *ECC*) are quite similar, but *ECC* obtains the best ranking. The reason for such results is motivated by the *CC* approach itself. When the classifier is not perfect, *CC* is only able to provide good estimates for a small range of the prevalence, for the rest of the values the estimates are worse. In the best scenario, bagging may improve the accuracy of a single logistic regression model, but it cannot correct the weakness of the *CC* approach. This behavior shall be analyzed graphically in the next section.

These results can be analyzed in different ways from an statistical point of view. First of all, and following [44], we have performed an statistical comparison in two steps: (i) a Friedman test rejects the hypothesis of all the methods performing at the same level and (ii) pairwise comparisons using the Bergmann-Hommel test with $\alpha = 0.05$ indicate that *EHDy* and *EAC* are significantly better than *CC* and *AC*. The difference between *EHDy* and *HDy* is not significant though. Also between *EAC* and *AC(bag)* and between *EHDy* and *HDy(bag)*. However, it is important to note that all these comparisons are deeply influenced by the inclusion of *CC* approaches, a group of algorithms that systematically perform worse than the other methods, and espe-

Table 3: Mean squared error using Naïve Bayes as base classifier. The score of the best performer in a group for each dataset is in bold

dataset	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
balance.1	0.0288	**0.0090**	0.0210	0.0135	0.0044	**0.0025**	0.0031	**0.0016**	0.0017
balance.2	0.2758	0.2758	**0.0979**	0.2758	0.2758	**0.0935**	0.2758	0.2758	**0.2753**
balance.3	0.0297	**0.0085**	0.0208	0.0082	0.0037	**0.0020**	0.0023	0.0019	**0.0014**
breast-cancer	0.0007	0.0007	**0.0006**	0.0007	0.0006	**0.0005**	0.0005	0.0005	**0.0004**
cmc.1	0.0672	0.0786	**0.0498**	0.0451	0.0266	**0.0137**	0.0147	0.0101	**0.0097**
cmc.2	0.1100	0.1223	**0.0431**	0.0307	0.0299	**0.0126**	0.0126	0.0120	**0.0109**
cmc.3	0.3391	0.3391	**0.0641**	0.3391	0.3391	**0.0485**	0.0211	0.0186	**0.0147**
ctg.1	0.0118	0.0119	**0.0073**	0.0026	0.0024	**0.0018**	0.0023	**0.0016**	**0.0016**
ctg.2	0.0125	0.0133	**0.0089**	0.0018	**0.0014**	0.0015	0.0022	**0.0017**	0.0018
ctg.3	0.0115	0.0148	**0.0052**	0.0035	0.0045	**0.0030**	0.0048	0.0045	**0.0036**
diabetes	0.0318	0.0360	**0.0261**	0.0133	0.0144	**0.0076**	0.0109	**0.0090**	**0.0090**
german	0.0852	0.0898	**0.0396**	0.0704	0.0196	**0.0096**	0.0166	0.0099	**0.0094**
haberman	0.2257	0.2895	**0.0752**	0.2689	0.2746	**0.0573**	0.0725	0.0856	**0.0562**
ionosphere	0.0119	0.0126	**0.0105**	0.0052	0.0055	**0.0047**	0.0054	**0.0053**	0.0054
iris.2	0.0054	0.0069	**0.0043**	0.0066	0.0078	**0.0034**	0.0035	0.0029	**0.0026**
iris.3	0.0099	**0.0044**	**0.0044**	0.0134	0.0047	**0.0040**	0.0052	0.0044	**0.0040**
mammographic	0.0148	0.0145	**0.0137**	0.0054	0.0056	**0.0047**	0.0046	**0.0034**	0.0037
pageblocks.5	**0.0437**	0.0530	0.0897	0.0162	**0.0125**	0.0905	**0.0120**	0.0146	0.1049
phoneme	**0.0173**	0.0178	0.0278	0.0012	0.0016	**0.0011**	**0.0010**	**0.0010**	0.0012
semeion.8	0.0098	0.0099	**0.0072**	0.0033	**0.0026**	0.0028	0.0046	0.0044	**0.0030**
sonar	**0.0223**	0.0234	0.0236	0.0235	0.0253	**0.0147**	0.0196	**0.0141**	0.0173
spambase	0.0072	0.0067	**0.0062**	**0.0004**	0.0005	**0.0004**	**0.0004**	**0.0004**	**0.0004**
spectf	0.0363	0.0341	**0.0316**	0.0391	0.0356	**0.0295**	0.0416	0.0394	**0.0271**
tictactoe	0.1083	0.0684	**0.0619**	0.1266	**0.0322**	0.0336	0.0249	0.0181	**0.0171**
transfusion	0.0703	0.1410	**0.0519**	0.0398	0.1165	**0.0265**	0.0316	0.0262	**0.0203**
wdbc	0.0027	0.0025	**0.0022**	0.0018	0.0017	**0.0012**	0.0014	**0.0013**	0.0014
wine.1	0.0011	0.0013	**0.0010**	**0.0010**	0.0014	**0.0010**	**0.0009**	0.0013	**0.0009**
wine.2	**0.0016**	0.0020	0.0018	**0.0014**	0.0021	0.0015	**0.0017**	0.0020	**0.0017**
wine.3	0.0010	0.0009	**0.0008**	0.0012	0.0009	**0.0007**	0.0009	**0.0005**	0.0006
wine-quality-red	0.0251	**0.0229**	0.0262	0.0078	0.0061	**0.0053**	**0.0050**	0.0062	0.0051
wine-quality-white	0.0367	**0.0355**	0.0367	0.0031	0.0032	**0.0022**	0.0024	**0.0021**	0.0024
yeast	0.0935	0.1018	**0.0385**	0.0269	0.0169	**0.0068**	0.0122	0.0077	**0.0075**
Average	0.0546	0.0578	0.0281	0.0437	0.0400	0.0153	0.0193	0.0184	0.0194
Average ranking	7.4531	7.6562	6.3125	5.6875	5.6562	2.6562	4.0312	3.1406	2.4062

cially because there exist dependencies and correlations between the studied methods. Table 5 contains the complete results of the Bergmann-Hommel test comparing *EHDy* and EAC with the rest of the methods considering both MAE and MSE.

Having in mind that our objective was to compare the ensemble adapted versions to its original counterparts, Wilcoxon signed-rank test seems more appropriate for the experiments performed. This test reveals that EAC performs significantly better than AC ($p = 0.0001$), with the same behavior being found for *EHDy* with respect to *HDy* ($p = 0.0013$). Table 6 shows the p-values for the comparison with all the methods.

Table 4: Mean squared error using SVM with RBF kernel as base classifier. The score of the best performer in a group for each dataset is in bold

dataset	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
balance.1	**0.0000**	0.0006	0.0003	0.**0001**	0.0005	0.0002	**0.0002**	0.0003	**0.0002**
balance.2	**0.2701**	0.2758	0.0337	0.3064	0.2758	**0.0456**	0.0713	0.1328	**0.0613**
balance.3	**0.0001**	0.0003	0.0003	**0.0001**	0.0002	0.0002	**0.0001**	0.0003	**0.0001**
breast-cancer	0.0006	0.0006	**0.0005**	0.0005	**0.0004**	**0.0004**	0.0004	**0.0003**	**0.0003**
cmc.1	0.0444	0.0447	**0.0392**	**0.0063**	0.0083	0.0099	0.0104	**0.0100**	0.0158
cmc.2	0.2283	0.2047	**0.0495**	0.0504	0.0461	**0.0147**	0.0216	0.0206	**0.0164**
cmc.3	0.1195	0.1328	**0.0581**	0.0404	0.0415	**0.0236**	0.0197	**0.0180**	0.0209
ctg.1	0.0103	0.0170	**0.0082**	**0.0020**	0.0034	0.0063	0.0073	0.0048	**0.0041**
ctg.2	0.0265	0.0416	**0.0111**	**0.0025**	0.0060	0.0097	0.0315	0.0186	**0.0161**
ctg.3	0.0142	0.0288	**0.0064**	**0.0045**	0.0051	0.0049	0.0200	0.0151	**0.0055**
diabetes	0.0488	0.0524	**0.0309**	0.0144	0.0161	**0.0134**	0.0406	**0.0174**	0.0265
german	0.0804	0.0797	**0.0344**	0.0174	0.0206	**0.0095**	0.0136	**0.0115**	0.0131
haberman	0.1971	0.1992	**0.0684**	0.1289	0.1304	**0.0402**	0.0714	0.0742	**0.0547**
ionosphere	**0.0031**	0.0043	0.0032	0.0024	0.0026	**0.0023**	0.0027	0.0033	**0.0022**
iris.2	0.0030	0.0032	**0.0026**	0.0032	0.0039	**0.0022**	0.0039	0.0039	**0.0018**
iris.3	0.0018	0.0051	**0.0016**	0.0018	0.0049	**0.0011**	0.0019	0.0033	**0.0012**
mammographic	**0.0118**	0.0131	0.0129	**0.0038**	0.0042	0.0040	0.0036	**0.0034**	0.0035
pageblocks.5	0.2900	0.2645	**0.1735**	0.2476	**0.1576**	0.1819	**0.1743**	0.2410	0.2019
phoneme	0.0094	0.0117	**0.0076**	**0.0005**	**0.0005**	0.0016	0.0024	**0.0016**	0.0026
semeion.8	**0.0055**	0.0113	0.0641	**0.0007**	0.0019	0.0621	0.0081	**0.0058**	0.1110
sonar	**0.0106**	0.0113	0.0107	0.0123	0.0144	**0.0079**	0.0083	**0.0060**	0.0105
spambase	**0.0034**	**0.0034**	0.0049	0.0007	**0.0005**	0.0012	0.0015	**0.0010**	0.0020
spectf	0.1213	0.1309	**0.0529**	0.1473	0.1796	**0.0394**	0.1025	0.0820	**0.0508**
tictactoe	**0.0004**	0.0009	0.0007	0.0004	0.0004	**0.0003**	**0.0005**	0.0006	0.0006
transfusion	0.3301	0.2884	**0.0646**	0.2981	0.1254	**0.0520**	0.0812	0.0762	**0.0573**
wdbc	**0.0033**	**0.0033**	0.0043	**0.0021**	0.0022	0.0023	0.0133	0.0131	**0.0076**
wine.1	**0.0107**	0.0297	0.0298	**0.0150**	0.0302	0.0206	0.0078	0.0108	**0.0074**
wine.2	0.0283	0.0291	**0.0240**	0.0195	0.0220	**0.0179**	0.0234	0.0271	**0.0231**
wine.3	0.0530	0.0793	**0.0339**	**0.0296**	0.0369	0.0305	0.0623	0.0540	**0.0507**
wine-quality-red	0.0215	**0.0207**	0.0291	0.0079	0.0073	**0.0051**	0.0053	**0.0047**	0.0066
wine-quality-white	0.0561	0.0567	**0.0357**	**0.0018**	0.0023	0.0027	0.0040	**0.0027**	0.0068
yeast	0.1087	0.1129	**0.0358**	0.0118	0.0131	**0.0050**	0.0080	**0.0063**	0.0068
Average	0.0660	0.0674	0.0292	0.0431	0.0364	0.0193	0.0257	0.0272	0.0247
Average ranking	6.4531	8.1094	5.8438	3.9219	4.7031	2.7656	4.8906	4.5469	3.7656

Table 3 reports the experimental results when Naïve Bayes is used as probabilistic classifier. On average it seems that these results are a little bit worse than those obtained using logistic regression, but the conclusions are similar. However, there are two interesting changes. In this case, *ECC* performs much better than *CC* and *CC(bag)*. It also occurs in the comparison between *EAC* and *AC(bag)*; now the difference is significant.

Studying the results of ensemble versions, they again outperform single quantifiers. On the one hand, *EAC* wins 28 times, *AC* wins just 2 and there are 2 ties. The difference between *EHDy* vs. *HDy* is less pronounced, with 23 victories for *EHDy*, 6 ties and 3 victories for *HDy*. Analyzing the

Table 5: Average ranking for all methods using different performance measures. Symbols § y † indicate a significant difference ($p < 0.05$) between *EAC* and *EHDy* and the corresponding method using a Bergmann-Hommel test, respectively

LR	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
MSE	7.1875 §†	7.2500 §†	7.0625 §†	5.5469 §†	5.3125 †	3.5156	3.7500	3.1719	2.2031
MAE	6.9219 §†	6.7500 §†	7.2812 §†	5.7344 §†	5.6875 §†	3.6094	3.5938	3.2344	2.1875
NB	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
MSE	7.4531 §†	7.6562 §†	6.3125 §†	5.6875 §†	5.6562 §†	2.6562	4.0312	3.1406	2.4062
MAE	7.1562 §†	7.0000 §†	6.5312 §†	5.9531 §†	5.8125 §†	2.9531	3.9688	2.9688	2.6562
SVM	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
MSE	6.4531 §†	8.1094 §†	5.8438 §†	3.9219	4.7031 §	2.7656	4.8906 §	4.5469	3.7656
MAE	6.0938 §†	7.8594 §†	6.2500 §†	4.0938	4.7344	2.9375	4.7188	4.4375	3.8750

results statistically using Bergmann-Hommel test, we can see that *EAC* is significantly better than *CC* and its counterpart *AC*, while *EHDy* is also significantly better than these two methods, but it is not with respect to *HDy*. Applying a Wilcoxon signed-rank test, we found that *EAC* performs significantly better than *AC* ($p = 0.00002$), and also *EHDy* with respect to *HDy* ($p = 0.0002$).

The situation changes when using SVM with RBF kernel as base classifier, see the scores in Table 4. The ensemble versions still perform better than their counterparts; *EAC* vs. *AC*: 17 wins, 0 ties and 15 losses; *EHDy* vs. *HDy*: 20 wins 2 ties and 10 losses. However, statistical tests reveal that the differences are no longer significant. In the *EHDy*-*HDy* comparison, test results are near to be significant though ($p = 0.078$). In our opinion, this behavior is explained by two facts: (i) SVM with RBF kernel are complex models and the risk of overfitting increases, and (ii) SVM is not originally a probabilistic classifier, with the probabilities resulting from an output post-process. These circumstances have a negative impact on the stability of the results, leading to some models with a poor performance on one hand, and accurate models on the other, i.e. a greater model performance variance. Notice that *CC* and *CC(bag)* with SVM-RBF are the worst performing algorithms by far considering the three base learners.

Table 5 contains a summary of the obtained results for all the base classifiers containing both the mean squared error (MSE), and the mean absolute error (MAE). Interestingly, *EAC* and *EHDy* rank most of the times in the two first positions. The only exception is when logistic regression is the base learner. The reason is because *HDy* algorithm clearly outperforms *AC*

Table 6: The two algorithms proposed, *EAC* and *EHDy*, are compared statistically with the rest of the methods using Wilcoxon sign-rank tests. The table shows the p-value of each comparison

LR		CC	CC(bag)	ECC	AC	AC(bag)	HDy	HDy(bag)
MSE	EAC	4.097e-08	8.338e-07	1.495e-06	0.0001433	0.0011340	0.4564072	0.2536064
	EHDy	1.287e-06	1.429e-07	3.174e-05	4.392e-05	5.642e-05	0.0013742	0.0018011
MAE	EAC	6.519e-09	8.847e-09	4.417e-06	0.0002057	0.0006889	0.2099543	0.1497597
	EHDy	1.727e-07	5.122e-08	6.379e-08	1.119e-05	7.716e-06	0.0012973	0.0016667
NB		CC	CC(bag)	ECC	AC	AC(bag)	HDy	HDy(bag)
MSE	EAC	7.517e-06	7.517e-06	2.011e-06	2.716e-05	7.653e-05	0.0083313	0.7420193
	EHDy	9.761e-06	4.209e-07	6.045e-05	7.935e-05	4.050e-05	0.0002547	0.0110552
MAE	EAC	1.287e-06	1.733e-06	5.122e-08	1.263e-05	4.570e-05	0.0803250	0.7719339
	EHDy	1.733e-06	3.512e-06	7.716e-06	2.924e-05	2.479e-05	0.0002778	0.1206512
SVM		CC	CC(bag)	ECC	AC	AC(bag)	HDy	HDy(bag)
MSE	EAC	3.736e-05	6.302e-06	2.692e-05	0.1322391	0.0152215	0.0038533	0.0152188
	EHDy	6.136e-05	2.312e-06	0.0148032	0.4388932	0.4890153	0.0786387	0.1681790
MAE	EAC	5.433e-05	5.871e-07	6.905e-07	0.1396090	0.0341199	0.0038655	0.0146764
	EHDy	0.0001472	3.512e-06	0.0058128	0.6378833	0.6511483	0.2241815	0.4773475

in that case. There are significant differences for Naïve Bayes and logistic regression, both for MSE and MAE scores.

5.3. Graphical analysis

An additional experiment was performed in order to analyze graphically the behavior of the quantifiers. The goal was to study the performance at specific prevalence values. The differences with respect to the first experiment are twofold: the number of subsamples generated for each test set, 210 instead of 100, and the values of the prevalence used. In this case, only 21 different values were considered, those nearest to [0:0.05:1]. Notice that sometimes it is impossible to obtain some of these values depending of the number of examples in the test set. In that case, the nearest possible value is used. Hence, there are 100 results for each prevalence (10 folds and 10 subsamples per fold, 210 divided by 21 prevalence values). The next figures represent in different ways such results.

Figure 6 shows the bias error (defined as $p-p'$) of *CC*, *AC*, *HDy*, *EAC* and *EHDy* over four datasets. Apparently, it seems that the overall performance of *AC*, *HDy*, *EAC* and *EHDy* is quite good in terms of bias: the average value predicted is near the true prevalence. But, notice that negative and positive biases cancel each other in the computation of bias, thus the graphics

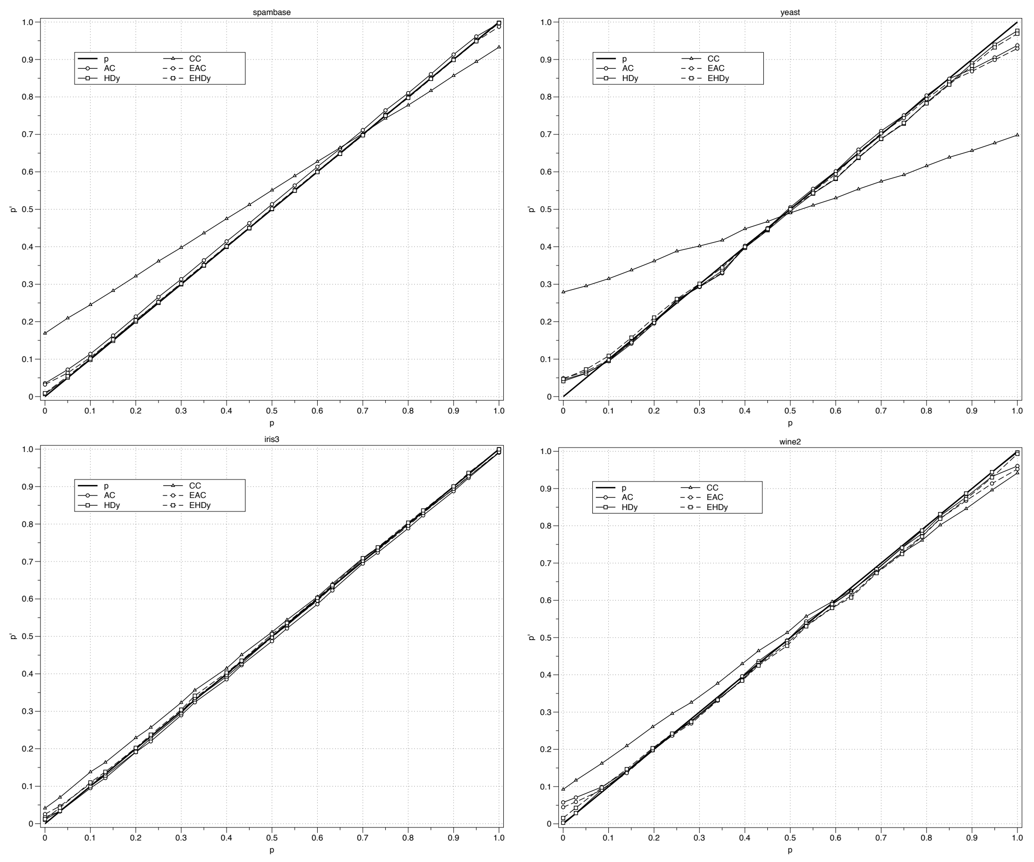
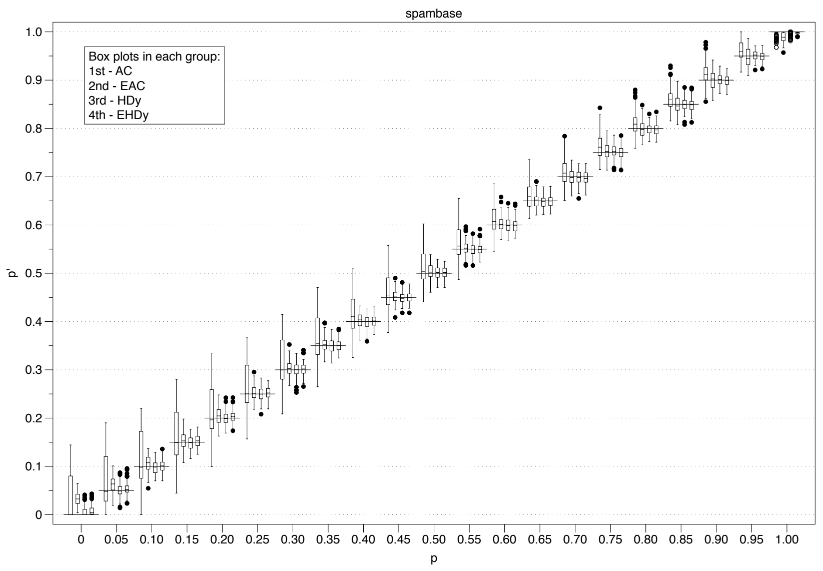


Figure 6: Analysis of bias when the prevalence varies in [0:0.05:1]. The figure compares five algorithms: *CC*, *AC*, *EAC*, *HDy* and *EHDy*. Each result comes from the prediction average of 100 sample sets with the same prevalence

do not show the magnitude of the errors in both directions.

However, bias graphics allow us to observe two important facts. First, *CC* follows the exact behavior analyzed theoretically by Forman in [17]: *CC* attains a perfect quantification at one point of p, and from there, *CC* underestimates the prevalence when p increases, and overestimates the prevalence when p decreases from *CC*'s optimal point. The deviation with respect to the true prevalence depends on the classifier accuracy. If the classifier is perfect, the bias is 0. But the bias increases when the accuracy decreases. This occurs for *spambase* and *yeast* datasets. The bias is smaller in the case of *wine2*, and tiny for *iris3*. The big issue of *CC* is that it is almost imposible



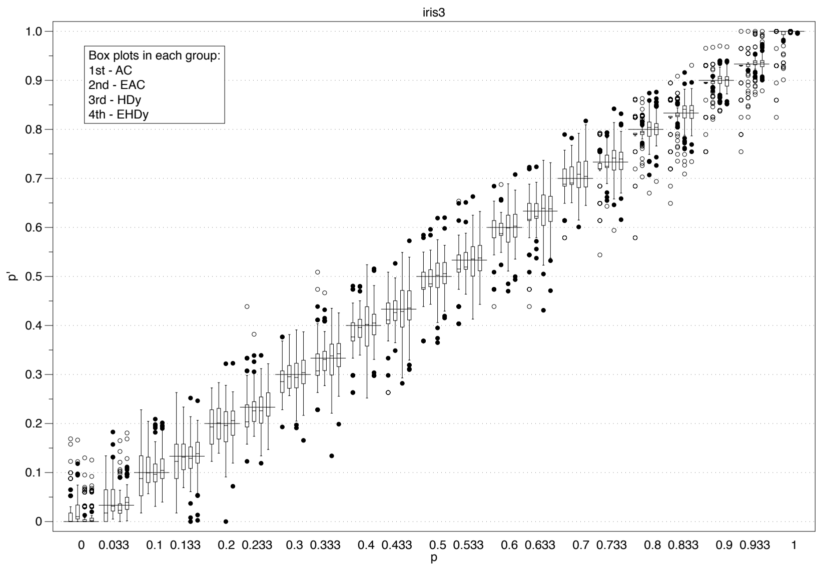
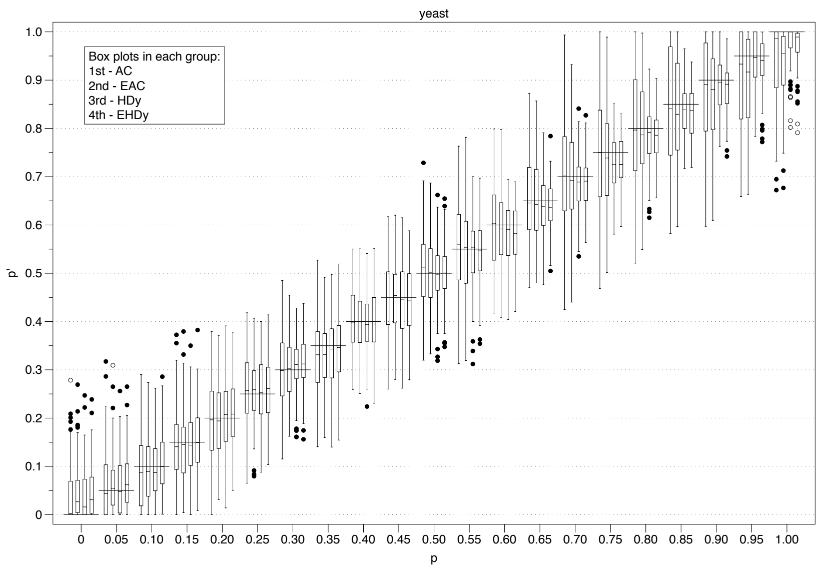


Figure 7: The figure shows the distribution of the predictions for *spambase* (39% of positives in the training set) and *iris3* (33%) datasets using a box plot. The horizontal lines represent the true prevalence for each group. Only four methods are displayed: *AC*, *EAC*, *HDy* and *EHDy*. Such methods appear always in the same order for each prevalence: first *AC*, then *EAC*, *HDy* is the third method and the last one is *EHDy*



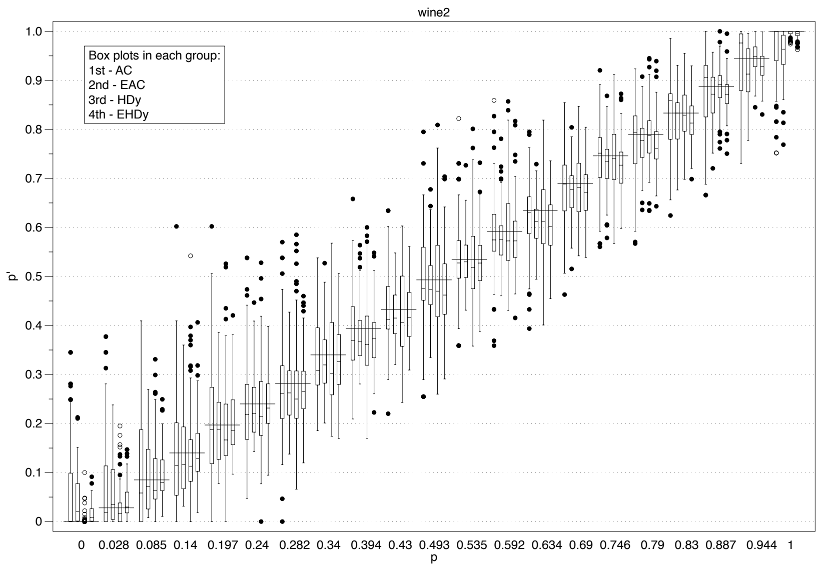


Figure 8: The figure shows the distribution of the predictions for *yeast* (29% of positives in the training set) and *wine2* (40%) datasets using a box plot. The horizontal lines represent the true prevalence for each group. Only four methods are displayed: *AC*, *EAC*, *HDy* and *EHDy*. Such methods appear always in the same order for each prevalence: first *AC*, then *EAC*, *HDy* is the third method and the last one is *EHDy*

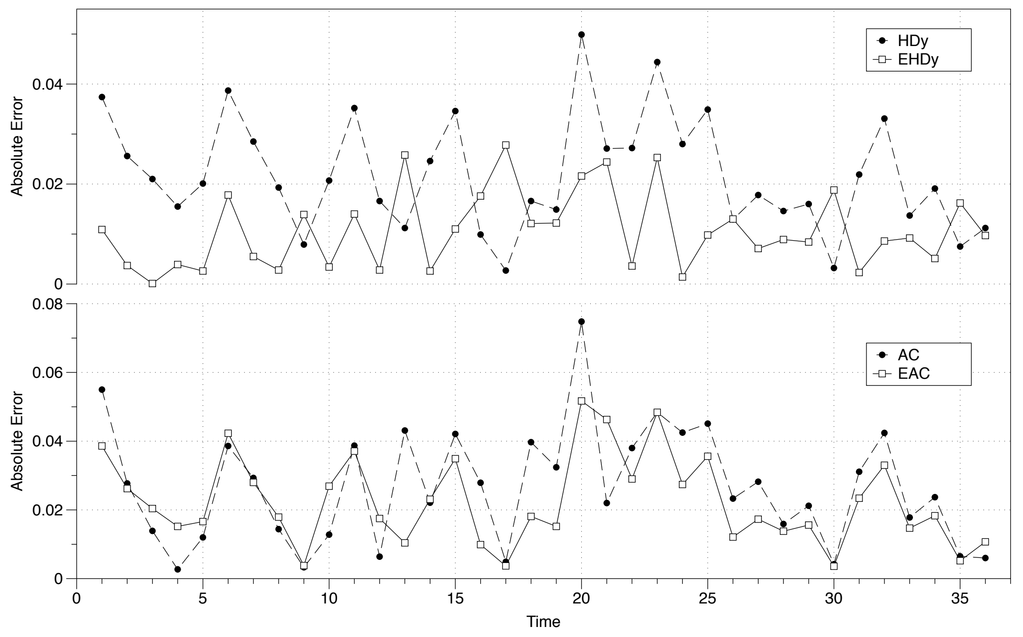


Figure 9: Results for Sentiment140 dataset. The figure shows the absolute error of *HDy* vs *EHDy* (top) and *AC* vs *EAC* (bottom)

to obtain perfect classifiers for real applications. The other interesting aspect of the bias analysis is that the ensemble versions *EAC* and *EHDy* have more bias than *AC* and *HDy*, mainly in the tails. This is in part due to the use of the mean as the aggregation strategy.

Figure 7 and Figure 8 represent the distribution of the predictions using box plots. Each box plot comprises a group of 100 predicted test sets for each prevalence. These graphics demonstrate that the quantification problem is more difficult that it seems at first glance. The errors are lower in the case of *spambase* and *iris3* datasets (Figure 7), but quite big for *yeast* and *wine2* datasets (Figure 8). The most interesting fact in these graphics is that the variability of *EAC* and *EHDy* is lower than that of *AC* and *HDy*. Notice that the body of the boxplot is usually smaller, and also they often have shorter whiskers and less outliers. For instance, these aspects can be easily observed in the case of *EAC* vs. *AC* over *spambase* dataset. This result is theoretically expected due to the use of ensembles. It is well known that ensembles tends to reduce the variance of the underlying classifier.

5.4. Twitter dataset

In order to test the behavior of the proposed approach over real data, we compare *EAC* and *EHDy* with their counterparts *AC* and *HDy* over Sentiment140 dataset [45]. This dataset is composed by $1,600,000$ Twitter messages with emoticons collected from April 6, 2009 to June 16, 2009. The tweets were labeled using emoticons: for instance, :-) in a tweet indicates that it contains a positive comment, while :-(indicates that the tweet expresses a negative sentiment. Tweets are represented using a bag-of-words approach, after having deleted the emoticons used for labelling purposes. The prevalence of the positive tweets vary between 56.21% and 64.41%, excluding the last day in which the prevalence is just 35.84%.

The base classifier employed was just logistic regression due to the size of the input space, most appropriate for linear classifiers. The experimental setup to train the models was exactly the same than that used for the previous experiments, described in Section 5.1. The quantification models are trained using the tweets from the two first days. Then, such models are employed to quantify the prevalence of the positive tweets for the rest of the days, representing a quite realistic experiment for quantification learning. Figure 9 shows the results in terms of the absolute difference between the estimated prevalence and the actual prevalence of each day.

Despite the prevalence of the positive comments varies slightly during the period, the performance of the ensemble versions is promising. *EAC* produces better predictions than *AC* in 24 out of 37 days, but the differences are not significant using a Wilcoxon sign rank test (p=0.07). In the comparison between *EHDy* and *HDY*, the differences are greater and significant (p=0.00016) in favor of the ensemble algorithm. The prevalence predicted by *EHDy* is more accurate for 30 days.

6. Conclusions

In this paper we have studied how ensembles behave in a problem characterized by the assumption of data distribution changing between training and test phases. Our core idea is to take advantage of that assumption and to use it in order to appropriately introduce diversity into the ensemble; we generate different training samples with each one representing a particular expected distribution change. We have experimentally applied our idea to the binary quantification problem, which is characterized by class prevalence

$\mathbf{P}(y)$ changing, but $\mathbf{P}(\boldsymbol{x}|y)$ remaining constant. Training samples are generated under this assumption, and as a result, the ensemble meta model is better suited for dealing with unseen prevalence test sets. Experimental results demonstrate that the ensemble quantifier adapted versions outperform its original counterparts.

One major contribution of this work is to propose a reasonable approach for using ensembles in quantification learning that performs better than state-of-the-art quantifiers. However, we also claim that the significance of this paper is not only limited to presenting the first ensemble-based quantifiers, but to introduce an idea that is applicable to other learning problems in which it is possible to define how the data distribution changes throughout the time. Moreover, we think that this work opens research problem within this type of learning tasks, in aspects like diversity inclusion in function of the expected changes, and model combining strategies. Although we have used a simple aggregation function, the prevalence mean, more complex strategies can be designed by taking into account additional information. For instance, the training prevalence of each individual ensemble model, or the similarity of its distribution with respect to the test set distribution. Both aspects represent interesting lines of research towards new ensemble learning studies.

Acknowledgments

This research has been funded by MINECO (the Spanish Ministerio de Economía y Competitividad) and FEDER (Fondo Europeo de Desarrollo Regional), grant TIN2015-65069-C2-2-R (MINECO/FEDER). Juan José del Coz is also supported by the Fulbright Commission and the Salvador de Madariaga Program, grant PRX15/00607.

References

[1] E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: Bagging, boosting, and variants, Machine Learning 36 (1-2) (1999) 105–139.

[2] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.

[3] Y. Freund, R. E. Schapire, et al., Experiments with a new boosting algorithm, in: ICML, vol. 96, 148–156, 1996.

[4] D. Opitz, R. Maclin, Popular ensemble methods: An empirical study, Journal of Artificial Intelligence Research (1999) 169–198.

[5] G. Fumera, F. Roli, A theoretical and experimental analysis of linear combiners for multiple classifier systems, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (6) (2005) 942–956.

[6] J. Quiñonero Candela, M. Sugiyama, A. Schwaighofer, N. D. Lawrence, Dataset shift in machine learning, The MIT Press, 2009.

[7] A. Storkey, When training and test sets are different: characterizing learning transfer, Dataset Shift in Machine Learning (2009) 3–28.

[8] M. Kull, P. Flach, Patterns of dataset shift, in: First International Workshop on Learning over Multiple Contexts (LMCE) at ECML-PKDD, 2014.

[9] J. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. Chawla, F. Herrera, A unifying view on dataset shift in classification, Pattern Recognition 45 (1) (2012) 521–530.

[10] T. G. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization, Machine Learning 40 (2) (2000) 139–157.

[11] L. I. Kuncheva, C. J. Whitaker, Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy, Machine Learning 51 (2) (2003) 181–207.

[12] R. E. Banfield, L. O. Hall, K. W. Bowyer, W. P. Kegelmeyer, Ensemble diversity measures and their application to thinning, Information Fusion 6 (1) (2005) 49–62.

[13] G. Brown, J. Wyatt, R. Harris, X. Yao, Diversity creation methods: a survey and categorisation, Information Fusion 6 (1) (2005) 5–20.

[14] S. Wang, X. Yao, Diversity analysis on imbalanced data sets by using ensemble models, in: Computational Intelligence and Data Mining, 2009. CIDM '09. IEEE Symposium on, 324–331, 2009.

[15] L. Kuncheva, Classifier ensembles for changing environments, in: Multiple Classifier Systems, Springer, 1–15, 2004.

[16] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Machine Learning 23 (1) (1996) 69–101.

[17] G. Forman, Quantifying counts and costs via classification, Data Mining and Knowledge Discovery 17 (2) (2008) 164–206.

[18] A. Esuli, F. Sebastiani, Sentiment Quantification, IEEE Intelligent Systems 25 (4) (2010) 72–75.

[19] L. Tang, H. Gao, H. Liu, Network quantification despite biased labels, in: Proceedings of the 8th Workshop on Mining and Learning with Graphs, ACM, 147–154, 2010.

[20] L. Sánchez, V. González-Castro, E. Alegre-Gutiérrez, R. Alaiz-Rodríguez, Classification and quantification based on image analysis for sperm samples with uncertain damaged/intact cell proportions, in: Proceedings of the 5th International Conference on Image Analysis and Recognition, ICIAR'08, 827–836, 2008.

[21] G. Forman, E. Kirshenbaum, J. Suermondt, Pragmatic text mining: minimizing human effort to quantify many issues in call logs, in: Proceedings of ACM SIGKDD'06, ACM, 852–861, 2006.

[22] D. Hand, Classifier technology and the illusion of progress, Statistical Science 21 (1) (2006) 1–14.

[23] W. Gao, F. Sebastiani, Tweet Sentiment: From Classification to Quantification, in: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, ACM, 97–104, 2015.

[24] T. Rakthanmanon, E. Keogh, S. Lonardi, S. Evans, MDL-based time series clustering, Knowledge and Information Systems 33 (2) (2012) 371–399.

[25] G. Forman, Counting positives accurately despite inaccurate classification, in: Machine Learning: ECML 2005, Springer, 564–575, 2005.

[26] V. González-Castro, R. Alaiz-Rodríguez, E. Alegre, Class Distribution Estimation based on the Hellinger Distance, Information Sciences 218 (2013) 146–164.

[27] L. Minku, A. White, X. Yao, The Impact of Diversity on Online Ensemble Learning in the Presence of Concept Drift, IEEE Transactions on Knowledge and Data Engineering 22 (5) (2010) 730–742, ISSN 1041-4347.

[28] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, F. Petitjean, Characterizing Concept Drift, Accepted for publication in Data Mining and Knowledge Discovery. arXiv preprint arXiv:1511.03816 .

[29] T. Fawcett, P. Flach, A response to Webb and Ting's on the application of ROC analysis to predict classification performance under varying class distributions, Machine Learning 58 (1) (2005) 33–38.

[30] I. Žliobaitė, Learning under Concept Drift: an Overview, Tech. Rep., Faculty of Mathematics and Informatics, Vilnius University, Lithuania, 2010.

[31] M. J. Hosseini, Z. Ahmadi, H. Beigy, Using a classifier pool in accuracy based tracking of recurring concepts in data stream classification, Evolving Systems 4 (1) (2013) 43–60.

[32] H. Wang, W. Fan, P. S. Yu, J. Han, Mining concept-drifting data streams using ensemble classifiers, in: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 226–235, 2003.

[33] A. Tsymbal, M. Pechenizkiy, P. Cunningham, S. Puuronen, Dynamic integration of classifiers for handling concept drift, Information fusion 9 (1) (2008) 56–68.

[34] M. Karnick, M. Ahiskali, M. D. Muhlbaier, R. Polikar, Learning concept drift in nonstationary environments using an ensemble of classifiers based approach, in: Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, IEEE, 3455–3462, 2008.

[35] K. O. Stanley, Learning concept drift with a committee of decision trees, Technical Report: UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin, USA .

[36] J. Z. Kolter, M. A. Maloof, Dynamic weighted majority: An ensemble method for drifting concepts, Journal of Machine Learning Research 8 (2007) 2755–2790.

[37] M. Scholz, R. Klinkenberg, Boosting classifiers for drifting concepts, Intelligent Data Analysis 11 (1) (2007) 3–28.

[38] J. J. Gart, A. A. Buck, Comparison of a Screening Test and a Reference Test in Epidemiologic Studies II. A Probabilistic Model for the Comparison of Diagnostic Tests, American Journal of Epidemiology 83 (3) (1966) 593–602.

[39] L. Milli, A. Monreale, G. Rossetti, F. Giannotti, D. Pedreschi, F. Sebastiani, Quantification trees, in: IEEE International Conference on Data Mining (ICDM'13), 528–536, 2013.

[40] J. Barranquero, P. González, J. Díez, J. J. Del Coz, On the study of nearest neighbor algorithms for prevalence estimation in binary problems, Pattern Recognition 46 (2) (2013) 472–482.

[41] J. Barranquero, J. Díez, J. J. del Coz, Quantification-oriented learning based on reliable classifiers, Pattern Recognition 48 (2) (2015) 591–604.

[42] A. Bella, C. Ferri, J. Hernández-Orallo, M. J. Ramirez-Quintana, Quantification via probability estimators, in: IEEE International Conference on Data Mining (ICDM'10), 737–742, 2010.

[43] R. Fan, K. Chang, C. Hsieh, X. Wang, C. Lin, LIBLINEAR: A Library for Large Linear Classification, Journal of Machine Learning Research 9 (2008) 1871–1874.

[44] S. García, F. Herrera, An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2677–2694.

[45] A. Go, R. Bhayani, L. Huang, Twitter sentiment classification using distant supervision, CS224N Project Report, Stanford 1 (2009) 12.