The Metric Dilemma: Competence-Conscious Associative Classification *

Adriano Veloso[a], Mohammed Zaki[b], Wagner Meira Jr.[a] and Marcos Gonçalves[a]
[a] Computer Science Dept, Federal University of Minas Gerais, Brazil
{adrianov,meira,mgoncalv}@dcc.ufmg.br
[b] Computer Science Dept, Rensselaer Polytechnic Institute, Troy, USA
zaki@cs.rpi.edu

Abstract

The classification performance of an associative classifier is strongly dependent on the statistic measure or metric that is used to quantify the strength of the association between features and classes (i.e., confidence, correlation etc.). Previous studies have shown that classifiers produced by different metrics may provide conflicting predictions, and that the best metric to use is data-dependent and rarely known while designing the classifier. This uncertainty concerning the optimal match between metrics and problems is a dilemma, and prevents associative classifiers to achieve their maximal performance. This dilemma is the focus of this paper.

A possible solution to this dilemma is to learn the competence, expertise, or assertiveness of metrics. The basic idea is that each metric has a specific sub-domain for which it is most competent (i.e., it consistently produces more accurate classifiers than the ones produced by other metrics). Particularly, we investigate stacking-based meta-learning methods, which use the training data to find the domain of competence of each metric. The meta-classifier describes the domains of competence (or areas of expertise) of each metric, enabling a more sensible use of these metrics so that competence-conscious classifiers can be produced (i.e., a metric is only used to produce classifiers for test instances that belong to its domain of competence). We conducted a systematic evaluation, using different datasets and evaluation measures, of classifiers produced by different metrics. The result is that, while no metric is always superior than all others, the selection of appropriate metrics according to their competence/expertise (i.e., competence-conscious associative classifiers) seems very effective, showing gains that range from 7% to 26% when compared to the baselines (SVMs and an existing ensemble method).

*This research was sponsored by UOL (www.uol.com.br) through its UOL Bolsa Pesquisa program (grant number 20080131200100), and partially supported by CNPq, Capes, Finep, Fapemig, and by the projects 5S-VQ (CNPq grant number 551013/2005-2), INCTWeb (CNPq grant number 573871/2008-6), and InfoWeb (CNPq grant number 550874/2007-0).

1 Introduction

There are countless paradigms and strategies for devising a classifier. One of these strategies is to explore relationships, dependencies and associations between features and classes. Such associations are usually hidden in the training examples, and when uncovered, they may reveal important aspects concerning the underlying phenomenon that generated these examples. These aspects can be exploited for sake of prediction. This strategy has led to a new family of classifiers which are often referred to as associative classifiers. The models used by these classifiers are composed of rules $\mathcal{X} \rightarrow c$, indicating an association between a set of features $\mathcal{X}$ and a class c. Associative classification has shown to be valuable in many applications, including document categorization [20], Web ranking [21] etc.

Associations may be defined in many ways. Correspondingly, there are many statistic measures or metrics that express, in different perspectives, the strength of feature-class associations (i.e., confidence, correlation etc.). Some perspectives should be emphasized in some cases, but may not be desired in others. Thus, as expected, the competence of a metric is data-dependent, in the sense that some metrics are well suited for some classification problems, but not for others (that is, each metric has a particular domain for which it is more competent). Competent metrics are rarely known while devising the classifier, and this dilemma concerning the best match between metrics and problems prevents the full potential of associative classifiers.

Obviously, one possible solution to the metric dilemma is to find the domain of competence (or areas of expertise) for each metric, that is, subsets of examples for which a certain metric produces better classifiers than the others. Having this information would enable the assignment of competent associative classifiers to specific problems according to their competence/expertise [8, 14]. Hopefully, classification performance would be drastically boosted by taking advantage of consciously assigning metrics to specific subsets of instances (i.e., a domain of competence). This would

be great, except that there are several metrics, and numerous (unknown) characteristics affecting their corresponding competence, and finding such an invariant domain of competence for metrics seems to be practically unfeasible.

As an alternate approach to the metric dilemma, we propose to automatically extract the competence of each metric. Taking as a starting point a set of q accurate[1] and diverse[2] metrics ($m_1, m_2, \ldots, m_q$), a stacking-like[3] meta-learning[4] strategy [18, 22] is used to extract, from the training data, information regarding the competence of each metric. This information is then used to enhance the original training data. Specifically, it is explicitly indicated the metrics that correctly classify each example in the training data (i.e., using a cross-validation procedure). This additional information is used to produce a meta-classifier which has the ability to consciously decide the appropriate match between metrics and examples (i.e., the meta-classifier is a function mapping features to competent metrics). Then, for each test instance t, the meta-classifier is used to decide which is the most competent metric to be applied, according to their expertise. A specific classifier, $\mathcal{C}^t_{m_i}$, is finally produced, so that m_i is expected to be the most competent metric to classify instance t (i.e., t belongs to the domain of competence of metric m_i). The classifiers that are produced following this strategy are regarded as *competence-conscious* classifiers. We propose two competence-conscious classifiers, with the difference between them residing in the way they perform the analysis of the domains of competence (or areas of expertise). The first classifier performs a coarse-grained, class-centric analysis, in which the domain of competence of a metric is composed of classes for which it produces accurate classifiers. The other classifier performs a fine-grained analysis, in which the domain of competence of a metric is composed of examples for which it produces accurate classifiers.

To evaluate the effectiveness of competence-conscious associative classifiers, we performed a systematic set of experiments using the UCI datasets, as well as more complex datasets obtained from other real applications, such as document categorization and Web spam detection. Our results suggest that the more fine-grained the analysis of the domains of competence, the more effective is the final classifier. The results also show that the proposed competence-conscious classifiers are able to outperform the baselines (SVMs and existing ensemble methods), providing gains ranging from 7% to 26%. The specific contributions of this paper are:

- We present a comprehensive study of the competence of associative classifiers produced by different statistic metrics. We show that no metric is consistently better than the others for all problems. Further, we also show that traditional metrics, such as confidence, are just moderately competent for most of the problems investigated.
- We propose approaches to the metric dilemma in the context of associative classification, which are based on learning the domain of competence of metrics.
- We propose competence-conscious classifiers, which effectively combine classifiers produced by different metrics (an ensemble) using their domains of competence. All constituent classifiers are produced using the same rule set. The only difference between the base classifiers is the way they interpret the rules (each classifier employes a different metric). Thus, in contrast to other ensemble approaches, the cost of producing a competence-conscious associative classifier is roughly the same cost of producing a single associative classifier.
- We present a deep evaluation of the proposed competence-conscious classifiers, and we show (using a set of complex datasets) that they are able to provide expressive gains in classification performance.

The remaining of this paper is organized as follows. In Section 2 we discuss related work. Then, in Section 3, we introduce our associative classification technique. The metric dilemma, and competence-conscious associative classifiers (i.e., the ensemble of classifiers produced by different metrics), are presented in Section 4. In Section 5 we evaluate the proposed competence-conscious classifiers, and compare them against state-of-the-art SVMs and other existing ensemble techniques. Finally, in Section 6 we conclude the paper.

2 Related Work

The ultimate goal of a classifier is to achieve the best possible classification performance for the problem at hand. An ensemble is a collection of classifiers whose predictions are combined with the goal of achieving better performance than the constituent classifiers. There is a body of evidence suggesting that ensembles offer substantial advantages in enough situations to be regarded as a major advance in machine learning [9]. Also, there is a body of theory explaining why ensembles work.

A variety of ensemble methods has already been proposed. Well known methods include bagging [5], boosting [15], and stacking [22]. In the following, we will focus our attention on stacking methods, since the techniques proposed in this paper are mostly related to them. Stacking

[1] An accurate metric is one that produces a classifier that has an error rate of better than random guessing.

[2] Two metrics are diverse if they produce classifiers that misclassify different instances.

[3] Stacking is based on the idea that different classifiers provide different but complementary explanations of the data.

[4] Literally, meta-learning means learning how to learn.

is based on the idea that different classifiers provide different but complementary explanations of the data. Thus, the predictions of these different (base) classifiers provide novel information that can be used as meta-features to form a new training data. Then, a meta-classifier is built using this new training data, but instead of predicting the correct class for a given test instance, the meta-classifier predictcs the base classifier that is most likely to correctly predict the class for such instance. The obvious advantage, in this case, is that the errors of a base classifier can be counterattacked by the hits of others.

In this paper we exploit staking based meta-learning approaches to address an important issue in associative classification: the *metric dilemma*. Several statistic metrics can be used to estimate *feature-class* associations [11, 13, 17], but the most competent one is rarely known in advance. Thus, we propose to explore the diversity among classifiers that are produced using different statistic metrics to maximize the performance of the final classifier (which will be refereed as a competence-conscious associative classifier). The metric dilemma is challenging, and, as far as we know, this is the first attempt to integrate classifiers produced by different statistic metrics, in the context of associative classification.

The integration of classifiers using strategies related to stacking was largely explored [2, 8, 10, 14, 19]. In [2], the authors use a neural network to learn, from predefined meta-features (e.g., maximum confidence, average confidence, number of applicable rules etc.), how to weigh the rules using a single association metric (i.e., confidence). We believe that the work of Ortega et. al [14] is the closest to ours. They used a referee (which in our case is a meta-classifier) to indicate the best classifier to be applied for each example. The approach used to produce the referee (which is based on decision trees) is different to the approach we used to produce the meta-classifier. *In our experiments we performed a direct comparison between the competence-conscious classifiers proposed in this paper and the ensemble approach proposed in* [14].

Self-delegation [8] is another strategy for combining the predictions of different base classifiers, and thus it is also related to our work. The idea is that each base classifier chooses by itself which instances it can safely classify. This choice is based on the confidence in its prediction. A base classifier delegates the difficult or uncertain predictions to other classifiers. Clearly, this strategy produces classifiers which are exclusively defined in terms of the original features (no meta-features are generated). This simplicity may be desirable, but it may neglect important information associated with meta-features. *We show this by performing a direct comparison between self-delegating classifiers and competence-conscious classifiers.*

The advantages of the proposed techniques, when compared against other ensemble techniques, are manifold. First, the proposed techniques showed to be more accurate than other ensemble techniques for most of the problems investigated. Further, all base classifiers $\mathcal{C}_{m_1}, \mathcal{C}_{m_2} \ldots \mathcal{C}_{m_q}$ are produced on a single shot, that is, the same rule-set is used to produce all base classifiers (i.e., the cost of generating several base classifiers is roughly the same cost of generating only one of them). In this case, the only difference between the base classifiers resides in the metric that is used to interpret and weigh feature-class associations. The only additional overhead that is necessary to generate competence-conscious classifiers, comes from the cost of enhancing the training data to produce the meta-classifier.

3 Associative Classification

The classification problem is defined as follows. We have an input dataset called the *training data* (denoted as $\mathcal{D}$) which consists of instances composed of a set of l attribute-values $(a_1, a_2, \ldots, a_l)$ along with a special variable called the *class*. The set of all possible attribute-values is denoted as $\mathcal{A}$, while the class variable draws its value from a discrete set of classes $(c_1, c_2, \ldots, c_n)$. The training data is used to build a classifier that relates features (or attribute values) to the class variable. The *test instances* are a set of instances for which only the features are known while the class value is unknown. The classifier, which is a function from $\mathcal{A}$ to $\{c_1, c_2, \ldots, c_n\}$, is used to predict the class value for test instances (i.e., the classifier is a function which maps a set of features to one of the classes).

Associative classifiers exploit the fact that, frequently, there are strong associations between features and classes. Typically, such associations are expressed using rules of the form $\mathcal{X} \rightarrow c_i$, where $\mathcal{X} \subseteq \mathcal{A}$ and c_i is one of the classes. These rules are usually hidden in the training data, and when uncovered they can be combined in order to accurately map features to classes (i.e., the classification function is obtained by combining the information provided by these rules). In the following we will denote as $\mathcal{R}$ an arbitrary rule set extracted from $\mathcal{D}$. Similarly, we will denote as $\mathcal{R}_{c_i}$ an arbitrary rule set composed of rules of the form $\mathcal{X} \rightarrow c_i$, such that $\mathcal{R}_{c_i} \subseteq \mathcal{R}$.

Naturally, some rules in $\mathcal{R}$ represent stronger associations than others. A set of statistic metrics that quantify the strength of the association between $\mathcal{X}$ and c_i are used to compare the rules. Associative classifiers usually learn the classification function in two broad steps:

1. generate a rule set, $\mathcal{R}$, from $\mathcal{D}$
2. estimate the likelihood of class membership for each test instance, by combining the information provided by rules in $\mathcal{R}$

In this paper our focus is on an important challenge associated with step 2: provide an accurate estimate of

the likelihood of class membership. Specifically, given an instance t, we want to estimate the likelihood $\hat{p}(c_i|t)$ that t belongs to class c_i. Only rules $\mathcal{X} \rightarrow \{c_1, c_2 \ldots c_n\} \in \mathcal{R}$, such that $\mathcal{X} \subseteq t$ are used to estimate $\hat{p}(c_1|t), \ldots \hat{p}(c_n|t)$. Such rules are said to match t, and they form the rule set $\mathcal{R}^t$.

The likelihood of membership of an instance t is estimated by combining rules in $\mathcal{R}^t$=$\{\mathcal{R}^t_{c_1} \cup \mathcal{R}^t_{c_2} \cup \ldots \cup \mathcal{R}^t_{c_n}\}$. A simple (yet effective) probabilistic strategy is to interpret $\mathcal{R}^t$ as a poll, in which rule $\mathcal{X} \rightarrow c_i \in \mathcal{R}^t_{c_i}$ is a vote given by $\mathcal{X}$ for class c_i. The weight of a vote $\mathcal{X} \xrightarrow{m} c_i$ depends on the strength of the association between $\mathcal{X}$ and c_i, which is given by an association metric m. Weighted votes for class c_i are summed and then averaged by the total number of votes for this class, as expressed by function $s(c_i, t)$, shown in Equation 3.1 (where $m(r)$ is the metric value for rule r). As will be discussed in the next section, there are situations in which $m(r) < 0$, and thus a value z (which is the lowest score, that is, z=$s(c_j, t)|s(c_j, t) \leq s(c_i, t) \forall c_i$), is used to ensure that all scores are greater than or equal to 0.

$$s(c_i, t) = \frac{\sum_{r \in \mathcal{R}^t_{c_i}} m(r)}{\mid \mathcal{R}^t_{c_i} \mid} - z \tag{3.1}$$

The likelihood of membership of t to class c_i is expressed by the function $\hat{p}(c_i|t)$, shown in Equation 3.2 (thus, votes with high weights increase the likelihood of the corresponding class being the correct one, while votes with low weights reduce the likelihood of the corresponding class being the correct one). A higher value of $\hat{p}(c_i|t)$ indicates a higher likelihood of t to belong to class c_i. The class associated with the highest likelihood is finally predicted. As will be shown in Section 5, association metrics play a fundamental role in estimating the likelihood of class membership. However, the best-quality, most competent metric is data-dependent, and rarely known while devising the classifier[5].

$$\hat{p}(c_i|t) = \frac{s(c_i, t)}{\sum_{j}^{n} s(c_j, t)} \tag{3.2}$$

4 The Metric Dilemma

Selecting an appropriate association metric is a major issue while designing an associative classifier. Classifiers produced by different metrics often present different classification performance. Depending on the characteristics of the problem, some metrics may be more suitable than others. That is, a sub-domain may present properties that make a metric more suitable than others. This suggests that classifiers produced by a certain metric are only able to make reliable predictions over a subset of the entire domain space, which is the area of expertise, or domain of competence, of such metric. In this section we exploit the training data to learn the competence, or expertise of each metric. Then, a specific metric is used to produce a classifier for subproblems that belong to its domain of competence.

4.1 Association Metrics Next we present several metrics for measuring the strength of association between a set of features ($\mathcal{X}$) and classes ($c_1, c_2, \ldots, c_n$). Some of these metrics are popular ones in routine use [1, 17], while others were recently used in the context of associative classification [3]. These metrics interpret association using different definitions. We believe that these definitions are sufficiently different to indicate that the corresponding classifiers may present some diversity.

- Confidence (m_1) [1]: This metric measures the fraction of instances in $\mathcal{D}$ containing $\mathcal{X}$ that belong to c_i. It is the conditional probability of c_i being the correct class of instance t given that $\mathcal{X} \subseteq t$, as shown in Equation 4.3. Its value ranges from 0 to 1.

$$m_1 = p(c_i|\mathcal{X}) \tag{4.3}$$

- Added Value (m_2) [11]: This metric measures the gain in accuracy obtained by using rule $\mathcal{X} \rightarrow c_i$ instead of always predicting c_i, as shown in Equation 4.4. Negative values indicate that always predicting c_i is better than using the rule. Its value ranges from -1 to 1.

$$m_2 = p(c_i|\mathcal{X}) - p(c_i) \tag{4.4}$$

- Certainty (m_3) [13]: This metric measures the increase in accuracy between rule $\mathcal{X} \rightarrow c_i$ and always predicting c_i, as shown in Equation 4.5. It assumes values smaller than 1.

$$m_3 = \frac{p(c_i|\mathcal{X}) - p(c_i)}{p(\overline{c_i})} \tag{4.5}$$

- Yules'Q (m_4) and Yules'Y (m_5) [17]: These metrics are based on odds value, as shown in Equations 4.6 and 4.7, respectively. Their values range from -1 to 1. The value 1 implies perfect positive association between $\mathcal{X}$ and c_i, value 0 implies no association, and value -1 implies perfect negative association.

[5] We denote as $\mathcal{C}_{m_j}$ an associative classifier which applies m_j as the association metric in Equation 3.1

(4.6)
$$m_4 = \frac{p(\mathcal{X} \cup c_i)p(\overline{\mathcal{X} \cup c_i}) - p(\mathcal{X} \cup \overline{c_i})p(\overline{\mathcal{X}} \cup c_i)}{p(\mathcal{X} \cup c_i)p(\overline{\mathcal{X} \cup c_i}) + p(\mathcal{X} \cup \overline{c_i})p(\overline{\mathcal{X}} \cup c_i)}$$

(4.7)
$$m_5 = \frac{\sqrt{p(\mathcal{X} \cup c_i)p(\overline{\mathcal{X} \cup c_i})} - \sqrt{p(\mathcal{X} \cup \overline{c_i})p(\overline{\mathcal{X}} \cup c_i)}}{\sqrt{p(\mathcal{X} \cup c_i)p(\overline{\mathcal{X} \cup c_i})} + \sqrt{p(\mathcal{X} \cup \overline{c_i})p(\overline{\mathcal{X}} \cup c_i)}}$$

- Strength Score (m_6) [3]: This metric measures the correlation between $\mathcal{X}$ and c_i, but it also takes into account how $\mathcal{X}$ is correlated with the complement of c_i (i.e., $\overline{c_i}$), as shown in Equation 4.8. Its value ranges from 0 to ∞.

(4.8)
$$m_6 = \frac{p(\mathcal{X}|c_i)p(c_i|\mathcal{X})}{p(\mathcal{X}|\overline{c_i})}$$

- Support (m_7) [1]: This metric measures the fraction of instances in $\mathcal{D}$ covered by the rule $\mathcal{X} \rightarrow c_i$, as shown in Equation 4.9. Its value ranges from 0 to 1.

(4.9)
$$m_7 = p(\mathcal{X} \cup c_i)$$

- Weighted Relative Confidence (m_8) [13]: This metric trades off accuracy and generality, as shown in Equation 4.10. The first component is the accuracy gain that is obtained by using rule $\mathcal{X} \rightarrow c_i$ instead of always predicting c_i. The second component incorporates generality.

(4.10)
$$m_8 = (p(c_i|\mathcal{X}) - p(c_i))p(\mathcal{X})$$

Although we focus our analysis only on these metrics, the techniques to be introduced here are general and able to exploit any number of metrics transparently. Next we will discuss a simple approach to boost classification performance by exploiting associative classifiers produced by these metrics.

Self-Delegating Classifier (SDC) Equation 3.2 can be used to estimate the reliability of a prediction, and this information can be used to select the most reliable prediction from all involved classifiers. The process is illustrated in Algorithm 1. For a given test instance t, the selected class is the one which is associated with the highest likelihood $\hat{p}(c_i|t)$ amongst all classifiers $\mathcal{C}^t_{m_1}, \mathcal{C}^t_{m_2}, \ldots, \mathcal{C}^t_{m_q}$. The basic idea is to use the most reliable prediction (among the predictions performed by all classifiers) to select the class for t.

Although simple, SDC does not exploit the competence of each metric. In fact, each base classifier simply decides by itself the instances it will classify, not meaning that the select instances belong to its domain of competence.

Algorithm 1 Classifier based on Self Delegation of Metrics.

Require: The training data $\mathcal{D}$, and a test instance t
Ensure: The class for instance t

1: $\mathcal{R}^t \Leftarrow$ rules $\mathcal{X} \rightarrow c_i$ (with $1 \leq i \leq n$) extracted from $\mathcal{D}$ such that $\mathcal{X} \subseteq t$
2: produce different classifiers $\mathcal{C}^t_{m_1}, \mathcal{C}^t_{m_2}, \ldots, \mathcal{C}^t_{m_q}$, for instance t, using rules in $\mathcal{R}^t$
3: **return** the class associated with the highest likelihood of membership for t (i.e., Eq. 2), amongst all classifiers

4.2 Learning the Metric Competence The optimal match between metrics and problems is a valuable information. In this section we present an approach to estimate such matching. The proposed approach may be viewed as an application of Wolpert's stacked generalization [22]. From a general point of view, stacking can be considered a meta-learning method, as it refers to the induction of classifiers over inputs that are, in turn, the predictions of other classifiers induced from the training data.

Algorithm 2 Enhancing the Training Data with the Competence of each Metric.

Require: The original training data $\mathcal{D}$, and a cross-validation parameter k
Ensure: The enhanced training data $\mathcal{D}_e$

1: split $\mathcal{D}$ into k partitions, so that $\mathcal{D}$=$\{d_1 \cup d_2 \cup \ldots \cup d_k\}$
2: $\mathcal{D}_e \Leftarrow \emptyset$
3: **for each** partition d_i **do**
4: **for each** instance $t \in d_i$ **do**
5: $m \Leftarrow \emptyset$
6: $\mathcal{R}^t \Leftarrow$ rules $\mathcal{X} \rightarrow c_i$ (with $1 \leq i \leq n$) extracted from $\{\mathcal{D}\text{-}d\}$ such that $\mathcal{X} \subseteq t$
7: produce different classifiers, $\mathcal{C}^t_{m_1}, \mathcal{C}^t_{m_2}, \ldots, \mathcal{C}^t_{m_q}$, using rules in $\mathcal{R}^t$
8: **for each** classifier $\mathcal{C}^t_{m_j}$ **do**
9: **if** $\mathcal{C}^t_{m_j}$ correctly predicts the class for t **then**
10: $m \Leftarrow m \cup m_j$
11: **end if**
12: **end for**
13: $\mathcal{D}_e \Leftarrow \mathcal{D}_e \cup \{t \cup m\}$
14: **end for**
15: **end for**

The process starts by enhancing the original training data using the outputs of the base classifiers, $\mathcal{C}^t_{m_1}$, $\mathcal{C}^t_{m_2} \ldots \mathcal{C}^t_{m_q}$. Algorithm 2 shows the basic steps involved in the process. Initially, the enhanced training data, $\mathcal{D}_e$ is empty. An example t, along with the competence of each metric with regard to t (i.e., which metric correctly predicted the class for t), is inserted into $\mathcal{D}_e$. The process continues until all examples are processed. In the end, for each example

Id	Class	Attribute-Values $a_1\ a_2 \ldots a_l$
1	c_1	1 3 ... 6
2	c_1	1 3 ... 7
3	c_1	2 4 ... 6
4	c_2	2 4 ... 7
5	c_2	2 5 ... 8
6	c_2	2 4 ... 6
7	c_3	1 3 ... 9
8	c_3	2 5 ... 9
9	c_3	2 4 ... 8
10	c_3	2 4 ... 9

Table 1: Training Data, $\mathcal{D}$.

Id	Class	Attribute-Values $a_1\ a_2 \ldots a_l$	Competent Metric(s) (per instance)	Most Competent Metric(s) (per class)
1	c_1	1 3 ... 6	m_2	m_1
2	c_1	1 3 ... 7	m_1 m_3	
3	c_1	2 4 ... 6	m_1	
4	c_2	2 4 ... 7	m_1 m_2	m_1
5	c_2	2 5 ... 8	m_1 m_2 m_3	
6	c_2	2 4 ... 6	m_1	
7	c_3	1 3 ... 9	m_2	m_2
8	c_3	2 5 ... 9	m_2 m_3	
9	c_3	2 4 ... 8	m_1 m_2 m_3	
10	c_3	2 4 ... 9	m_2	

Table 2: Enhanced Training Data, $\mathcal{D}_e$.

$t \in \mathcal{D}_e$ we have a list of metrics that produced a competent classifier for t, and this information enables learning the comains of competence of each metric, as will be discussed in the next section.

To illustrate this process, please consider the example shown in Tables 1 and 2. Table 1 shows the original training data, $\mathcal{D}$. Using the process described in Algorithm 2, the competence of each metric to each instance is appended to $\mathcal{D}$, resulting in the enhanced training data, $\mathcal{D}_e$, which is shown in Table 2. In this case, for a given example t, metric m_i is shown if the corresponding classifier $\mathcal{C}^t_{m_i}$ has correctly classified t using the stacking procedure (i.e., metric m_i is competent with regard to example t). The enhanced training data, $\mathcal{D}_e$, can be exploited in several ways. In particular, we will use $\mathcal{D}_e$ to produce competence-conscious classifiers, as will be discussed next.

4.3 Competence-Conscious Classifiers In this section we present strategies for exploiting $\mathcal{D}_e$ in order to produce competence-conscious classifiers. The challenge, in this case, is to properly select a competent metric for a specific problem. The competence-conscious classifiers to be presented differ in how they perform the analysis of the domains of competence of metrics.

Class-Centric Competence-Conscious Classifier (C^5) The competence of a metric is often associated with certain classes. Some metrics, for instance, produce classifiers which show preference for more frequent classes, while others produce classifiers which show preference for less frequent ones. As an illustrative example, please consider Table 2. Metric m_1 is extremely competent for classifying instances that belong to classes c_1 and c_2. On the other hand, if we consider instances belonging to c_3, metric m_2 perfectly classifies all instances. This information (which is shown in the last column of Table 2) may be used to produce class-centric competence-conscious classifiers. The process is depicted in Algorithm 3. It starts with a meta-classifier, $\mathcal{M}$, which learns the most competent metric for a given class. Any classifier can be used to build the meta-classifier. For simplicity we choose an associative classifier that weights the votes given by rules using the confidence metric. In this case, instead of generating rules $\mathcal{X} \rightarrow c_i$, the meta-classifier generates rules $\mathcal{X} \rightarrow m_i$, which maps features (i.e., in the second column of Table 2) to metrics (i.e., in the fourth column of Table 2). Then, for each test instance t, the meta-classifier indicates the most competent metric, m_j, that is then used to produce the final classifier, $\mathcal{C}^t_{m_j}$, which is finally used to predict the class for instance t.

Algorithm 3 Class-Centric Meta Classifier.

Require: The enhanced training data $\mathcal{D}_e$ (i.e., the 3^{rd} and 5^{th} columns of Table 2), and a test instance t

Ensure: The most competent metric for instance t

1: **for each** metric m_i **do**
2: $\mathcal{R}^t_{m_i} \Leftarrow$ rules $\mathcal{X} \rightarrow m_i$ extracted from $\mathcal{D}_e$ such that $\mathcal{X} \subseteq t$
3: Estimate $\hat{p}(m_i|t)$, according to Equation 3.2 (using confidence to weigh the votes)
4: **end for**
5: **return** metric m_j such that $\hat{p}(m_j|t) > \hat{p}(m_i|t) \forall i \neq j$

Instance-Centric Competence-Conscious Classifier (IC^4) Although the competence of some metrics are associated with certain classes, specific instances may be better classified using other metrics. In such cases, a more fine-grained analysis of competence is desired. As an illustrative example, please consider again Table 2. Although metric m_1 is the most competent one to classify instances belonging to class c_1, metric m_2 is the only one which competently classifies instance 1 (which belongs to c_1). Again, a meta-classifier, $\mathcal{M}$, is used to explore such cases. The process is depicted in algorithm 4. In this case, the meta-classifier

learns the most competent metric by generating rules of the form $\mathcal{X} \rightarrow m_i$, which maps features (i.e., the second column of Table 2) to metrics (i.e., in the third column of Table 2). Then, for each test instance t, the meta-classifier indicates the most competent metric, m_j, which is used to produce the final classifier, $\mathcal{C}^t_{m_j}$.

The main advantage of C^5 and IC^4 is that, in practice, multiple metrics produce competent classifiers for a particular instance t, but $\mathcal{M}$ needs to predict only one of them (competent metrics are not mutually exclusive, and thus, in practice, multiple metrics produce competent classifiers for t). This redundancy in competence that exists when different metrics are taken into account, may increase the chance of selecting a competent metric.

Algorithm 4 Instance-Centric Meta Classifier.

Require: The enhanced training data $\mathcal{D}_e$ (i.e., the 3^{rd} and 4^{th} columns of Table 2), and a test instance t

Ensure: The most competent metric for instance t

1: **for each** metric m_i **do**
2: $\mathcal{R}^t_{m_i} \Leftarrow$ rules $\mathcal{X} \rightarrow m_i$ extracted from $\mathcal{D}_e$ such that $\mathcal{X} \subseteq t$
3: Estimate $\hat{p}(m_i|t)$, according to Equation 3.2 (using confidence to weigh the votes)
4: **end for**
5: **return** metric m_j such that $\hat{p}(m_j|t) > \hat{p}(m_i|t) \forall i \neq j$

Algorithm 5 Competence-Conscious Classifiers.

Require: The training data $\mathcal{D}$, the meta-classifier $\mathcal{M}$, and a test instance t

Ensure: The class for instance t

1: **for each** class c_i **do**
2: $\mathcal{R}^t_{c_i} \Leftarrow$ rules $\mathcal{X} \rightarrow c_i$ extracted from $\mathcal{D}$ such that $\mathcal{X} \subseteq t$
3: **end for**
4: select the most competent classifier for t, $\mathcal{C}^t_{m_x}$, using $\mathcal{M}$
5: Estimate $\hat{p}(c_i|t)$ (with $1 \leq i \leq n$), according to Equation 3.2 (using metric m_x to weigh the votes)
6: **return** class c_j such that $\hat{p}(c_j|t) > \hat{p}(c_i|t) \forall i \neq j$

Bounds for Competence-Conscious Classifiers We derived lower and upper bounds for the classification performance of the proposed competence-conscious associative classifiers. The lower bound is the performance that is obtained by randomly selecting a competent metric. Clearly, this lower bound increases with the redundancy between the base classifiers, $\mathcal{C}^t_{m_1}, \ldots \mathcal{C}^t_{m_q}$ (this redundancy exists because competent metrics are not mutually exclusive, and, thus, for a particular instance t, multiple metrics can be competent). The upper bound is the classification performance that would be obtained by an oracle which always predicts a competent metric (note that perfect performance is not always possible, since it may not exist a competent metric for some instances). Clearly, this upper bound increases with the accuracy and diversity associated with base classifiers.

5 Experimental Evaluation

In this section we will empirically analyze the proposed classifiers, SDC, C^5, and IC^4. In our experiments, we used 26 datasets from the UCI Machine Learning Repository [4] and two datasets obtained from more complex applications. These datasets cover a wide range of properties. We compare the proposed classifiers against SVM [12] baselines[6], and against the ensemble approach proposed in [14], which we call ER[7] (standing for External Referee). For associative classifiers, continuous attributes in the training data were discretized using the entropy-minimization method [7], and the attribute-values in the test set were simply mapped to the corresponding intervals (in this way, the discretization process did not use class information in the test set). Experiments that compare classification performance report results for the standard 10-fold cross-validation procedure. In all experiments, parameter k for Algorithm 2 was set to 2 (i.e., each training data, $\mathcal{D}$, was splited in two disjoint partitions, d_1 and d_2, in order to obtain the enhanced training data, $\mathcal{D}_e$). Best results, including statistical ties, are emphasized. A bold face indicates that the corresponding result was found statistically significant at the 95% confidence level when tested with the two-tailed paired t-test. Experiments were run on 1.8 MHz Intel processors 1GB RAM under Linux.

5.1 Document Categorization The first dataset was extracted from the first level of the ACM Computing Classification System (`http://portal.acm.org/dl.cfm/`). The dataset contains 6,682 documents labeled using the 8 first level categories of ACM, namely Hardware (C1), Computer Systems Organization (C2), Software (C3), Computing Methodologies (C4), Mathematics of Computing (C5), Information Systems (C6), Theory of Computation (C7), Computing Milieux (C8). Citations and words in title/abstract compose the set of features. The dataset has a vocabulary of 9,840 unique words, and a total of 51,897 citations. Please, refer to [20] for a detailed description of this dataset.

Using the rules extracted from this dataset, we can analyze the relationship between the widely used confidence metric (m_1) with other metrics, as shown in Figure 1 (to ease the observation of this relationship, we also include, in each

[6] We used the LibSVM tool [6] in order to select appropriate parameters, which are informed in each experiment.

[7] The ensemble is composed of the base classifiers $\mathcal{C}_{m_1}, \ldots, \mathcal{C}_{m_8}$, but the best classifier for each test instance is selected using a decision tree referee.

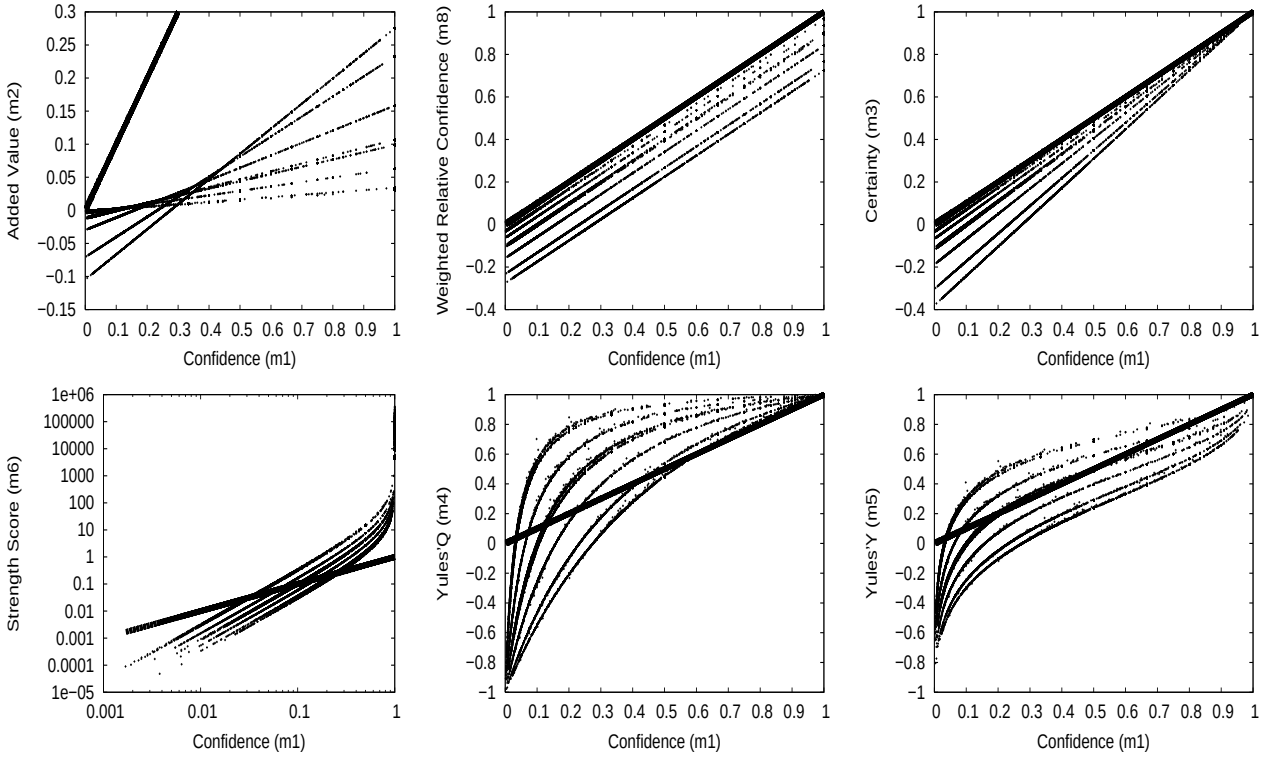

Figure 1: Relationship between Confidence and other Metrics using the ACM Dataset.

	$\mathcal{C}_{m_1}$	$\mathcal{C}_{m_2}$	$\mathcal{C}_{m_3}$	$\mathcal{C}_{m_4}$	$\mathcal{C}_{m_5}$	$\mathcal{C}_{m_6}$	$\mathcal{C}_{m_7}$	$\mathcal{C}_{m_8}$	Lower Bound	SDC	C[5]	IC[4]	Upper Bound	ER	SVM
C1	0.809	**0.846**	0.826	0.834	0.834	**0.848**	0.183	0.628	0.715	0.813	0.809	0.821	0.893	0.801	0.729
C2	0.714	0.785	0.758	0.772	0.799	0.752	0.313	0.785	0.723	0.730	0.738	0.766	0.880	0.719	**0.879**
C3	0.912	0.851	0.888	0.871	0.864	0.748	**0.960**	0.880	0.870	0.876	0.884	0.918	0.983	0.874	0.661
C4	0.569	**0.690**	0.628	0.657	0.661	0.676	0.090	0.547	0.562	0.581	0.623	0.623	0.795	0.604	0.515
C5	0.548	0.624	0.593	0.675	0.680	0.670	0.010	0.329	0.563	0.568	0.625	0.648	0.751	0.613	**0.907**
C6	**0.948**	0.929	**0.937**	**0.931**	0.927	0.893	0.689	0.761	0.877	0.919	0.911	0.925	0.965	0.898	0.869
C7	**0.922**	0.893	0.897	0.890	0.887	0.889	0.507	0.687	0.837	0.906	0.895	0.902	0.922	0.876	0.672
C8	0.641	0.715	0.687	0.721	0.729	**0.755**	0.071	0.481	0.591	0.654	0.697	0.697	0.823	0.674	**0.771**
Total	0.843	0.847	0.850	0.852	0.855	0.810	0.566	0.735	0.798	0.848	0.858	**0.881**	0.925	0.811	0.827

Table 3: Classification Performance associated with each Category of the ACM Dataset.

graph, a thicker line which indicates the corresponding confidence value). Each point in the graphs corresponds to a rule, for which it is shown the values of some metrics (i.e., confidence in the x-axis and another metric in the y-axis). Clearly, each metric has its particular behavior with varying values of confidence. We will use these relationships to understand some of the results to be presented. For lower values of confidence, Added Value (m_2) has a preference for less frequent classes, but, after a certain confidence value, the preference is for more frequent classes. Certainty (m_3) always prefer less frequent classes, but linearly approaches confidence as its value increases. Yules'Q (m_4) and Yules'Y (m_5) have a similar behavior, showing preference for less frequent classes and hardly penalizing associations with low confidence values. Strength Score (m_6) and Weighted Relative Confidence (m_3) both prefer less frequent classes, but Strength Score shows a non-proportional preference for associations with higher values of confidence. The relationship between confidence and support (m_7) is omitted, but, by definition, support shows a preference for more frequent classes. We will use these relationships to explain some of the results reported in the following[8].

[8] Due to lack of space, we only show the relationship between confidence and other metrics using this dataset, however, similar behaviors were observed in the other datasets.

Table 3 shows the classification performance obtained by different classifiers using the ACM dataset (for this application, performance is computed through the traditional accuracy). We will first analyze the performance associated with each category, and then the final classification performance, which is shown in the last line of the table. Classifiers produced by confidence ($\mathcal{C}_{m_1}$) and support ($\mathcal{C}_{m_7}$) performed very well in the most frequent categories (Software, Inf. Systems and Theory of CS). On the other hand, instances belonging to less frequent categories (Comp. Methodologies, Mathematics of CS, and CS Organization) were better classified using Yules'Q ($\mathcal{C}_{m_4}$) and Yules'Y ($\mathcal{C}_{m_5}$). This is expected, and is in agreement with the behaviors depicted in Figure 1 (Yules'Y and Yules'Q show a preference for less frequent categories). The best metric is the one that better balances its performance over all categories. Although the classifier produced by Yules'Y was not the best one for any specific category of ACM, it was the best overall classifier (amongst classifiers produced by other metrics in isolation).

SDC shows a performance that is similar to the performance obtained by most of the base classifiers (the improvement, when it exists, is only marginal). Competence-conscious classifiers C^5 and IC^4 showed the best performances. IC^4 outperformed all other classifiers, providing gains of more than 7%, when compared against SVM [9], and gains of more than 8.5% when compared against ER. IC^4 is always far superior than the corresponding lower bound, but it is also relatively far from the corresponding upper bound.

We also performed an analysis on how the different metrics were used by C^5 and IC^4, as can be seen in Figure 2 (Left). C^5 utilized only few metrics, specially m_2, m_3 and m_7. Metric m_4 was used to produce classifiers to only one category, and metrics m_5 and m_8 were not used (this is because these two metrics were not the most competent in any category of ACM, and therefore are not considered by C^5). IC^4, on the other hand, utilized all metrics, specially m_1, m_2 and m_3. Both C^5 and IC^4 make large utilization of metrics m_2 and m_3. For C^5, some areas of expertise can be easily detected. Metric m_2 is considered competent for categories Hardware and CS Organization, while metric m_3 is considered competent for category Information Systems. For IC^4, areas of expertise are finer grained, but with manual inspection we detected that m_1 is considered competent for category CS Organization, and m_3 is considered competent for category Milieux.

We finalize this first set of experiments by analyzing one of the reasons of the good performance showed by IC^4. Figure 2 (Right) shows the accuracy associated with scenarios for which a different number of metrics are competent. The frequency of occurrence of each scenario is also shown (note that both accuracy and frequency values are shown in the y-axis). As it can be seen, for more than 7% of the instances no metric is competent, and, obviously, these instances were misclassified (this means that the inclusion of other metrics may improve classification performance in this dataset). As expected, accuracy increases with the number of competent metrics. For almost half of the instances all 8 metrics are competent. In these scenarios, there is no risk of misclassification, since a classifier produced by any metric will perform a correct prediction. The accuracy associated with scenarios where only 7 and only 6 of the metrics are competent, is also extremely high (respectively, 99% and 96%). These three scenarios (i.e., 8, only 7, and only 6 metrics are simultaneously competent) correspond to 86% of the instances, and the average accuracy associated with these three scenarios is almost 98% for IC^4. Further, IC^4 shows to be more robust that C^5, providing superior accuracy (relative to the accuracy of C^5) in scenarios where there are only few competent metrics.

5.2 Web Spam Detection In this application the objective is to detect malicious actions aimed at the ranking functions used by search engines. We used a dataset obtained from the Web Spam Challenge (`http://webspam.lip6.fr/wiki/pmwiki.php`). The dataset is very skewed (only 6% of the examples are spam pages). Each example is composed of direct features (i.e., number of pages in the host, number of characters in the host name etc.) link-based features (i.e., in-degree, out-degree, PageRank etc.) and content-based features (i.e., number of words in the page, average word length etc.).

Table 4 shows the classification performance obtained by different classifiers (for this application, performance is computed through accuracy, F_1 measure[10], and the area under the curve). $\mathcal{C}_{m_1}$ and $\mathcal{C}_{m_7}$ showed impressive performance in terms of accuracy. This is expected, because the vast majority of examples are legitimate pages, and confidence and support have preference for more frequent classes. On the other hand, $\mathcal{C}_{m_1}$ and $\mathcal{C}_{m_7}$ showed poor performance in terms of F_1 and AUC (i.e., no spam pages were detected). The remaining base classifiers were able to detect some spam pages, specially $\mathcal{C}_{m_6}$, which also shows impressive performance in terms of accuracy. In terms of AUC, $\mathcal{C}_{m_2}$ and $\mathcal{C}_{m_3}$ showed the best performance, amongst base classifiers. Thus, different metrics show distinct performance depending to the evaluation target (i.e., accuracy, F_1 or AUC).

Now we evaluate C^5 and IC^4, which are the best performers in terms of F_1. Although IC^4 showed to be far from the optimal performance, it showed impressive gains when compared against SVM [11] and ER, in terms of F_1 and AUC.

[9] Polynomial kernel of degree 6.

[10] F_1 is a combination of precision (p) and recall (r) defined as their harmonic mean $\frac{2pr}{p+r}$

[11] Linear kernel with parameter C set to 5.00.

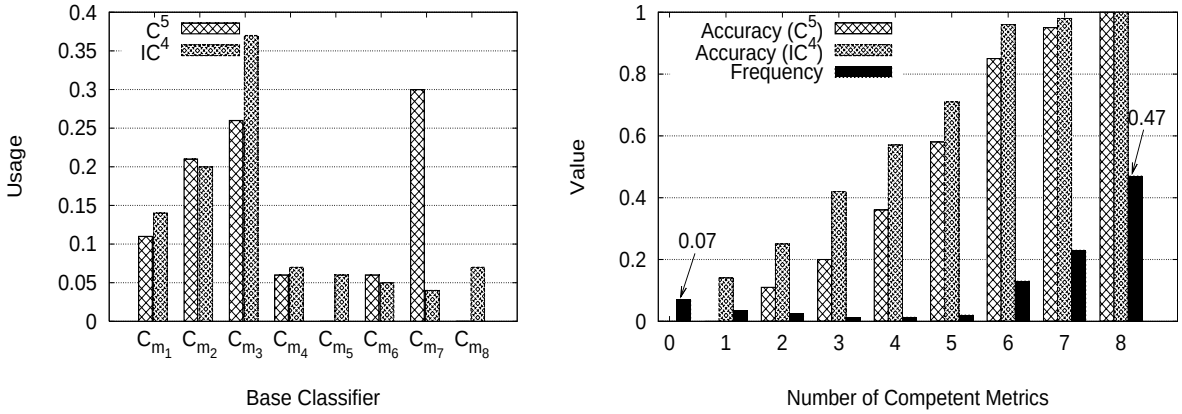

Figure 2: Left − Metric Utilization in the ACM Dataset. Right − Distribution of Competent Metrics in the ACM Dataset.

	$\mathcal{C}_{m_1}$	$\mathcal{C}_{m_2}$	$\mathcal{C}_{m_3}$	$\mathcal{C}_{m_4}$	$\mathcal{C}_{m_5}$	$\mathcal{C}_{m_6}$	$\mathcal{C}_{m_7}$	$\mathcal{C}_{m_8}$	Lower Bound	SDC	C^5	IC^4	Upper Bound	ER	SVM
Acc.	**0.946**	0.704	0.702	0.894	0.901	**0.948**	**0.946**	0.880	0.852	0.861	0.870	0.897	0.990	0.866	**0.956**
F_1	0.486	0.522	0.522	0.584	0.589	0.592	0.486	0.587	0.588	0.594	0.609	**0.624**	0.947	0.586	0.504
AUC	0.500	0.756	0.756	0.607	0.606	0.562	0.500	0.629	0.662	0.730	0.718	**0.789**	0.908	0.725	0.512

Table 4: Classification Performance for Web Spam Detection.

5.3 UCI Datasets In the last set of experiments, we used 26 datasets obtained from the UCI Machine Learning Repository [4]. Table 5 shows the performance obtained by each classifier using these datasets (for this application, performance is computed through the traditional accuracy). $\mathcal{C}_{m_4}$ and $\mathcal{C}_{m_5}$ showed poor performance in skewed datasets where few classes are much more frequent than the others (i.e., anneal, lymph, auto, hypo). This is because Yules'Y and Yules'Q have preference for less frequent classes (as shown in Figure 1). For these skewed datasets, $\mathcal{C}_{m_7}$ (support) showed its best performance, since the likelihood of predicting most frequent classes is higher in such datasets (this is expected, due to the definition of support). For most of the datasets, $\mathcal{C}_{m_1}$, $\mathcal{C}_{m_2}$ and $\mathcal{C}_{m_3}$ are in close rivalry ($\mathcal{C}_{m_1}$ shows a slightly better average performance, but $\mathcal{C}_{m_3}$ shows better performance more often). $\mathcal{C}_{m_6}$ (strength score) shows the best average performance, amongst all base classifiers.

On average, C^5 shows superior classification performance than SDC. Also, the performance of C^5 is, on average, slightly superior than the performance obtained by SVM[12] and ER baselines. Again, IC^4 is the best performer, and for some datasets it reaches a performance that is close to optimal (i.e., anneal, breast, hypo, iris, labor, sick, wave and wine), suggesting that the more fine-grained the analysis of competence, the more effectively the metrics are combined. Interestingly, the performance of C^5 approaches the performance of IC^4 for datasets containing more classes (i.e., glass, led7, lymph, vehicle, and zoo), since in this case the competence analysis performed by C^5 becomes finer grained.

Some datasets deserve special attention. IC^4 showed very good performance in the anneal dataset. Figure 3(left) shows the frequency distribution of competent metrics for this dataset. Almost 70% of the instances have more than five competent metrics, and in such scenarios accuracy reaches 100%. The accuracy obtained in such scenarios guarantees a final classification performance that is already superior than the performance of $\mathcal{C}_{m_2}$, $\mathcal{C}_{m_4}$, $\mathcal{C}_{m_5}$ and $\mathcal{C}_{m_8}$. Similar trends also happens in datasets austra, breast, cleve, german, heart, hypo, iono, iris, sick and wine. In the auto dataset IC^4 showed poor performance, being worse than base classifiers $\mathcal{C}_{m_1}$, $\mathcal{C}_{m_2}$, $\mathcal{C}_{m_3}$ and $\mathcal{C}_{m_6}$. Figure 3(right) shows the frequency/accuracy distribution of competent metrics for this dataset. As can be seem, the accuracy associated with almost 40% of the instances falls below 58%, which are the scenarios with less than 5 competent metrics. Also, we believe that, for such datasets, the meta-classifier was not able to correctly distinguish the domains of competence. Similar trend also happens for datasets hepati, tic-tac, and wave.

We finish our evaluation with simple linear models that are used to assess the improvements provided by C^5 and IC^4,

[12] Linear kernels were used for datasets austra (C=1.50), breast(C=3.00), cleve (C=3.00), crx (C=0.10), diabetes (C=1.00), german (C=1.00), heart (C=5.00), hepatitis (C=0.10), horse (C=1.00), hypo (C=3.00), ionosphere (C=0.50), led7 (C=0.50), pima (C=0.10), sick (C=5.00), sonar (C=5.00), and tic-tac-toe (C=0.50). Polynomial kernels were used for datasets anneal (degree=6), iris (degree=4), lymph (degree=5), vehicle (degree=6), waveform(degree=5), and wine(degree=5). RBF kernels were used for datasets auto (γ=0.00003), glass(γ=0.0012), led7(γ=0.0012), and zoo(γ=0.0012).

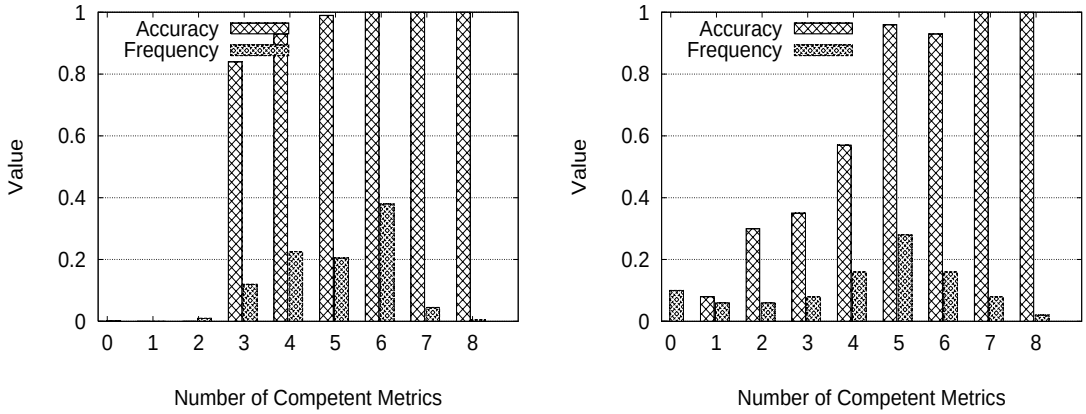

Figure 3: Distribution of Competent Metrics for IC^4 in the anneal (left) , and auto (right) Datasets.

	$\mathcal{C}_{m_1}$	$\mathcal{C}_{m_2}$	$\mathcal{C}_{m_3}$	$\mathcal{C}_{m_4}$	$\mathcal{C}_{m_5}$	$\mathcal{C}_{m_6}$	$\mathcal{C}_{m_7}$	$\mathcal{C}_{m_8}$	LB	SDC	C^5	IC^4	UB	ER	SVM
anneal	0.762	0.693	0.863	0.114	0.230	0.928	0.762	0.615	0.624	0.923	0.935	**0.961**	0.997	0.923	0.949
austra	0.857	0.850	0.850	0.857	0.850	0.855	0.830	0.860	0.855	**0.872**	**0.869**	**0.878**	0.923	0.848	0.857
auto	0.712	0.751	0.760	0.048	0.107	**0.778**	0.404	0.517	0.532	0.680	0.700	0.695	0.897	0.673	0.721
breast	0.941	**0.971**	**0.971**	**0.969**	**0.969**	**0.969**	0.929	**0.968**	0.951	0.955	**0.966**	**0.972**	0.984	**0.972**	**0.972**
cleve	**0.841**	0.831	0.831	0.834	0.834	0.818	0.838	0.828	0.835	**0.840**	0.836	**0.848**	0.901	**0.840**	0.834
crx	0.831	0.849	0.849	0.853	0.842	**0.857**	0.842	**0.859**	0.843	0.855	**0.863**	**0.870**	0.923	**0.861**	**0.856**
diabet	**0.781**	0.744	0.744	0.748	0.748	**0.781**	0.700	0.743	0.745	0.754	**0.781**	**0.785**	0.935	0.773	0.770
germa	0.700	0.693	0.693	0.691	0.693	**0.747**	0.700	0.726	0.721	0.723	**0.738**	**0.748**	0.952	**0.738**	0.712
glass	**0.714**	0.658	0.672	0.644	0.644	**0.710**	0.565	0.649	0.635	0.662	**0.704**	0.683	0.865	0.669	**0.709**
heart	0.818	0.840	0.840	0.825	0.829	0.833	0.829	0.833	0.826	0.834	0.844	**0.859**	0.900	**0.851**	0.838
hepati	0.800	0.774	0.780	0.838	0.832	0.845	0.793	**0.851**	0.793	0.804	0.832	0.839	0.987	0.820	0.813
horse	0.713	0.728	0.728	0.706	0.687	0.750	0.774	0.717	0.739	0.759	0.772	**0.812**	0.894	0.782	**0.822**
hypo	0.952	0.875	0.876	0.126	0.126	0.976	0.952	0.935	0.728	0.884	0.939	**0.996**	1.000	0.952	0.987
iono	0.900	0.894	0.891	0.877	0.868	**0.925**	0.692	0.843	0.857	0.898	0.914	**0.944**	0.983	0.920	0.917
iris	**0.940**	**0.946**	**0.946**	**0.946**	**0.946**	0.933	**0.940**	**0.940**	0.935	**0.944**	**0.948**	**0.951**	0.953	**0.944**	**0.957**
labor	**1.000**	0.947	0.929	0.754	0.894	0.947	0.631	0.929	0.965	0.971	**1.000**	**0.993**	1.000	0.978	0.782
led7	0.745	0.738	0.738	0.741	0.740	0.715	0.743	0.743	0.737	0.749	**0.770**	**0.762**	0.807	0.743	0.746
lymph	**0.858**	0.763	0.817	0.060	0.141	0.783	0.750	0.777	0.581	0.793	**0.841**	**0.841**	0.946	**0.830**	0.803
pima	0.733	0.744	0.744	0.748	0.748	0.781	0.691	0.743	0.742	0.754	0.767	**0.798**	0.940	0.771	0.770
sick	0.938	0.642	0.648	0.126	0.136	0.969	0.938	0.679	0.628	0.923	0.945	**0.983**	1.000	0.964	0.968
sonar	0.812	**0.865**	**0.865**	0.850	**0.865**	0.831	0.769	**0.860**	0.817	0.855	**0.870**	**0.868**	0.952	**0.868**	0.840
tic-tac	0.653	**0.911**	0.812	**0.926**	**0.926**	0.812	0.415	0.534	0.763	0.835	0.882	**0.917**	1.00	0.879	0.833
vehicle	0.655	0.666	0.666	0.669	0.653	0.702	0.534	0.618	0.655	0.668	**0.725**	**0.737**	0.762	0.701	**0.726**
wave	0.805	0.808	0.807	0.816	0.812	0.816	0.783	0.787	0.806	0.812	0.812	0.839	0.839	0.822	**0.868**
wine	0.913	0.932	0.932	0.820	0.831	**1.000**	0.685	0.747	0.831	0.883	**1.000**	**0.992**	1.000	0.943	0.979
zoo	0.845	0.851	0.871	0.861	0.782	0.910	0.712	0.693	0.822	0.899	**0.935**	**0.947**	0.970	0.952	0.931
Avg.	0.816	0.808	0.812	0.673	0.683	0.846	0.738	0.769	0.769	0.828	**0.860**	**0.865**	0.935	0.847	0.845

Table 5: Classification Performance of Classifiers in the UCI Datasets.

relative to the base classifiers (a similar evaluation approach was used in [16]). Specifically, we are interested in modeling the accuracy of competence-conscious classifiers using the best results obtained by the base classifiers. Thus, we assumed a linear relationship between the accuracy obtained by the best base classifier and the accuracy obtained by either C^5 or IC^4. We characterized this relation by using statistical correlation coefficients (CC).

The associated regression lines were built using the 26 UCI datasets (i.e., each point corresponds to one of the 26 datasets). Regression lines for C^5 and IC^4 are shown in Figure 4, and both have very high correlation coeficients (which are shown between parenthesis). Further, their regression gradients are higher than one, possibly indicating, in the limit, competence-conscious associative classifiers are indeed more accurate than the best base classifier.

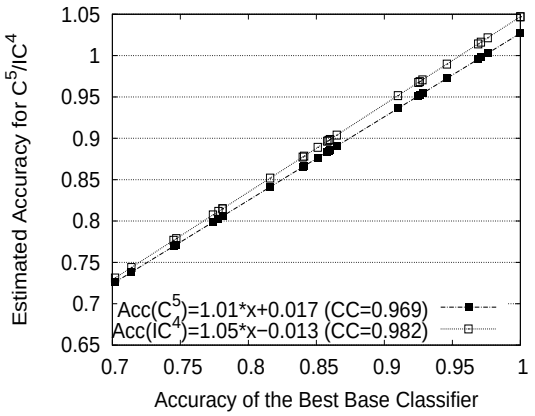

Figure 4: Accuracy Model for Competence-Conscious Classifiers.

6 Conclusions

This paper focused on an important problem in associative classification, the *metric dilemma*. We have shown that the performance of associative classifiers are strongly dependent on the metric that is used to quantify the strenght of the association between features and classes. There is no perfect metric, and no metric is consistently superior than all others, in the sense that it can be safely used in isolation. In fact, each metric has a particular domain of competence, or area of expertise, for which it is able to produce the most accurate classifier. We investigate meta-learning methods, which use the training data to learn the domain of competence of each metric. Finally, the competence of metrics are exploited to decide which is the best metric to be applied in each scenario, resulting in a combination or ensemble of classifiers produced by different metrics, which maximizes the performance of the final classifier, that we denoted as *competence-conscious* associative classifiers.

For effective metric combination, the corresponding classifiers must cover different portions of the training data (i.e., the metrics must show some diversity), and the training data must have features that are able to distinguish those portions of the training data. If these favorable conditions are met, our method reaches full potential of the base classifiers (i.e., the performance is close to the upper bound). On the other hand, a performance penalty may result.

We proposed competence-conscious classifiers, where the difference between them resides in how they perform the analysis of the domains of competence. The coarse-grained analysis, performed by the class-centric approach (C^5) provides lower gains when compared to the fine-grained analysis, performed by the instance-centric approach (IC^4), which outperforms all the evaluated classifiers, including a simple delegation approach (SDC), an existing ensemble method (ER), and SVMs.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, pages 207–216. ACM, 1993.

[2] M. Antonie, O. Zaïane, and R. Holte. Learning to use a learned model: A two-stage approach to classification. In *ICDM*, pages 33–42, 2006.

[3] B. Arunasalam and S. Chawla. CCCS: a top-down associative classifier for imbalanced class distribution. In *KDD*, pages 517–522. ACM, 2006.

[4] C. Blake and C. Merz. UCI repository of machine learning datasets. 1998.

[5] L. Breiman. Bagging predictors. *Mach. Learn.*, 24(2):123–140, 1996.

[6] C.-C. Chang and C.-J. Lin. *LibSVM: A Library for Support Vector Machines*, 2001. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[7] U. Fayyad and K. Irani. Multi interval discretization of continuous-valued attributes for classification learning. In *IJCAI*, pages 1022–1027. M. Kaufmann, 1993.

[8] C. Ferri, P. Flach, and J. Hernández-Orallo. Delegating classifiers. In *ICML*, page 37. ACM, 2004.

[9] J. Friedman. Comment on "Classifier Technology and the Illusion of Progress" by D. Hand. *Statistical Science*, 21(1):15–18, 2006.

[10] J. Gama and P. Brazdil. Cascade generalization. *Machine Learning*, 45:315–343, 2000.

[11] R. Hilderman and H. Hamilton. Evaluation of interestingness measures for ranking discovered knowledge. In *PAKDD*, pages 247–259. Springer, 2001.

[12] T. Joachims. Training linear SVMs in linear time. In *KDD*, pages 217–226. ACM, 2006.

[13] N. Lavrac, P. Flach, and B. Zupan. Rule evaluation measures: A unifying view. *Inductive Logic Prog.*, 1634:174–185, 1999.

[14] J. Ortega, M. Koppel, and S. Argamon. Arbitrating among competing classifiers using learned referees. *KAIS*, 3:470–490, 2001.

[15] R. Schapire. A brief introduction to boosting. In *IJCAI*, pages 1401–1406. M. Kaufmann, 1999.

[16] A. Seewald. Exploring the parameter state space of stacking. In *ICDM*, pages 685–688, 2002.

[17] P. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *KDD*, pages 32–41. ACM, 2002.

[18] K. Ting and I. Witten. Issues in stacked generalization. *Journal of Artificial Intelligence Res.*, (10):271–289, 1999.

[19] A. Tsymbal, M. Pechenizkiy, and P. Cunningham. Dynamic integration with random forests. In *ECML*, pages 801–808, 2006.

[20] A. Veloso, W. Meira, M. Cristo, M. Gonçalves, and M. Zaki. Multi-evidence, multi-criteria, lazy associative document classification. In *CIKM*, pages 218–227. ACM, 2006.

[21] A. Veloso, H. Mosrri, M. Gonçalves, and W. Meira. Learning to rank at query-time using association rules. In *SIGIR*, pages 267–274. ACM, 2008.

[22] D. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.