Social choice in distributed classification tasks: dealing with vertically partitioned data*

Mariana R. Mendoza and Ana L. C. Bazzan
PPGC, Instituto de Informática, UFRGS
Porto Alegre, RS, Brazil

Abstract

In many situations, a centralized, conventional classification task can not be performed because the data is not available in a central facility. In such cases, we are dealing with distributed data mining problems, in which local models must be individually built and later combined into a consensus, global model. In this paper, we are particularly interested in distributed classification tasks with vertically partitioned data, i.e., when features are distributed among several sources. This restriction implies a challenging scenario given that the development of an accurate model usually requires access to all the features that are relevant for classification. To deal with such a situation, we propose an agent-based classification system in which the preference orderings of each agent regarding the probability of an instance to belong to the target class are aggregated by means of social choice functions. We employ this method to classify microRNA target genes, an important bioinformatics problem, showing that the predictions derived from the social choice tend to outperform local models. This performance gain is accompanied by others interesting advantages: the combination methods herein proposed are extremely simple, do not require transfer of large volumes of data, do not assume an offline training process or parameters setup, and preserves data privacy.

Keywords: distributed data mining; vertical partition; heterogeneous sources; social choice theory

1 Introduction

A quite common assumption in classification tasks is that the data set for model development is centralized and fully accessible by the classifier [1, 39]. In this case, there are several well-established classification algorithms that are able to provide accurate models [17]. Nonetheless, in many

*This version is a preprint of the paper submitted to the journal Information Sciences

situations data centralization may be impracticable or undesirable due to context-specific constraints, e.g., storage and computing costs, communication overhead and privacy or intellectual property concerns [30], resulting in a distributed data mining (DDM) problem. Examples of scenarios where these restrictions may arise are biomedical research, fraud detection in financial organizations and calendar management by software assistants, in which ethical, legal or privacy issues prevent data sharing, thereby inducing a physical distribution of data. Under these constraints, data mining requires distributed data analysis, with minimal data communication among sources [20]. Generally, DDM is performed by generating local models based on the distributed data analysis and then adopting a strategy to combine them into a composite, global model [39].

In a typical DDM setting, data sets are partitioned primarily in one of two ways [20, 28], as shown in Figure 1. On the one hand, the sources may be homogeneous, in the sense that each site contains exactly the same features set, i.e., the same description, for different instances across similar domains. This is referred to as *horizontally partitioned data*. As an example, consider the case of several weather stations registering the same features sets (e.g., temperature, humidity, wind speed) to characterize weather in a geographical distributed environment (different instances), whose data will be further used for meteorological predictions. Despite the physical distribution of information and the impossibility to communicate the raw data given its prohibitively large volume [24], classical learning algorithms are more easily adaptable to this scenario given that all local models are learned from data of comparable quality and deal with the same form of input (the set of features), which facilitates their comparison and, eventually, their combination into a global model [40].

On the other hand, data sources may be heterogeneous, which means that they carry different kinds of information, i.e., different features sets, related to the same set of instances. In this scenario, data distribution occurs by means of *vertically partitioned data*. For instance, biomedical applications often need to consult records distributed among several heterogeneous domains, such as clinical data, genotype data and medical imaging, to define a more accurate diagnostic for a single patient. Under this condition, the distributed nature of data is a more critical issue because conventional classification algorithms are likely to fail in building a precise model given that the accurate prediction of the class of unlabeled instances usually requires access to all features that are relevant for their classification [28]. Moreover, the combination of locally developed models into a global model is not very straightforward because their performance can present a substantial variation for different parts of the input space and not every combination strategy can effectively deal with this situation [38].

As already noted in literature, problems related to DDM with homogeneous data sources have been widely studied and, in general, they are more

easily treatable with existing classification algorithms [40, 38, 21]. Conversely, the accurate classification under distributed, vertically partitioned data remains an open and challenging problem in the field [28, 39]. Hence, in this paper we are concerned in improving classification results in problems whose scenario implies a vertical partitioning of data among several heterogeneous sources. We assume that data centralization is neither possible nor desirable due to domain-specific constraints as aforementioned, and consequently a non-standard strategy must be applied to overcome this drawback.

A natural choice to deal with applications that require distributed problem solving are multiagent systems (MAS) [22, 34], which not only are inherently distributed systems, but also cope well with heterogeneous data. Indeed, there has been an increasing interest around the integration of agent technologies and data mining, as discussed in the recent publication by Cao and colleagues [6]. The approach proposed in the current work meets this trend by developing an agent-based data mining system to address vertically partitioned data in DDM. More precisely, agents in our system encapsulate distinct machine learning (ML) algorithms and are responsible for two main tasks, i) individually build local models based on their private information about the classification task, and ii) collaboratively work with other agents towards the construction of a global, consensus model. Here, we focus on a specific research question related to DDM [5], namely how to integrate the knowledge unveiled by a group of agents into a globally coherent model.

Typically, communication is a bottleneck in distributed applications because the cost of transferring large blocks of data may be too expensive and jeopardize the efficiency of the system [34]. Yet, neither simple combiners

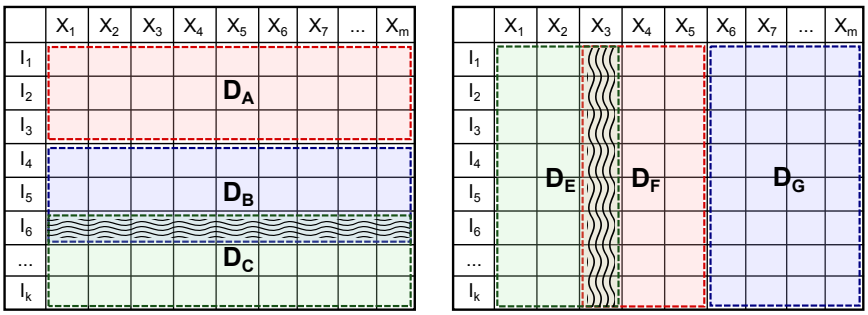


Figure 1: In DDM, data sets may be partitioned either a) horizontally, in which homogeneous sources D_A, D_B and D_C carry the same type of information (features set) about different instances, or b) vertically, in which the features set is distributed among heterogeneous sources D_E, D_F and D_G. Note that overlaps may occur in both situations (hachured area).

such as averaging, nor meta-learners, which usually require few data communication among sources, are considered suitable to deal with heterogeneous environments [38]. Simple combiners are vulnerable to the large discrepancies that may be observed among agents' performance over different parts of the input space. Meta-learners that assign different weights to agents according to the quality of their predictions often require off-line training, which may be unfeasible for a large and distributed data set, especially if the features are distributed. Here, we investigate the viability of mechanisms inspired by the social choice theory as a strategy to derive a consensus model in DDM problems, while efficiently handling the fundamental trade-off between communication and accuracy. Our approach does not require frequent communication or transferring of large blocks of data among agents and is also independent of parameters setting and tunning. Still, empirical evaluation shows that this apparently simple approach is rather robust and efficient to deal with DDM, being able to improve classification results in real-world DDM tasks even when the scenario is characterized by heterogeneous data sources. Therefore, another direct application of our agent-based system is the derivation of a "consensus" classifier model from an ensemble of classifiers, following the ensemble learning paradigm [10].

The remainder of this paper is organized as follows. In Section 2 we provide an overview of previous approaches for DDM problems, focusing on solutions that address vertically partitioned data and that preserve data privacy. Next, Section 3 and Section 4 present background on the methods, and the structure and functioning of the proposed agent-based system.Section 5 introduces the application in the domain of bioinformatics to which we apply the proposed system. Results drawn from empirical evaluation are presented in Section 6. Last, we discuss our findings and disclose final remarks in Section 7.

2 Related work

As pointed by Cao and colleagues [6], agent mining is an emerging interdisciplinary field that aims at the interaction and integration between multiagent systems and data mining and ML to deal with intrinsic challenges and needs faced by the constituent technologies. Many research papers have already proposed agent-based solutions for distributed classification tasks [19, 20, 35, 38, 22, 29, 1, 32, 40], some of which rely on collaborative learning, meaning that agents share information and perform negotiation among themselves while managing to devise a coherent global model. In what follows we review some of these efforts, with a special focus on applications addressing heterogeneous data domains with concerns regarding data privacy.

Modi and Shen [29] proposed two decentralized algorithms based on

collaborative learning for distributed classification tasks with vertically partitioned data. The core of their solution consisted of a learning process in which agents encapsulating the same algorithm build their models individually, based on their own private data, but collaborate with others by means of exchange of information in order to refine each others model. The information shared during the learning process consist of agents' classification model and/or the ids of the training instances, which does not compromise data privacy. At the end of the process, agents undergo simple voting to collectively decide the group's prediction. Authors argue that whenever synchronization is unfeasible, simply choosing some agent in advance as the designated predictor has comparable results in relation to the voting approach. Nonetheless, as we discuss in more details later, this strategy may not be suitable results when agents build their models using different learning or classification techniques.

Collaborative learning was also explored by Santana and colleagues [32] in the development of the NeurAge system, an agent-based neural system able to handle heterogeneous data sources. Although all agents encapsulate the same algorithm, i.e., a neural network, they differ in relation to the parameters used in their learning process. Learning occurs by agents communicating and negotiating among themselves in order to achieve a consensus classification when it comes to a new input, i.e., an unlabeled instance. Specifically, negotiation is performed in terms of agents' confidence about the class of a given input over several rounds, each of which is characterized by agents suggesting decreases in each others' confidence based on a sensitivity analysis. At the end of the process, the agent with the highest confidence is said to be the most appropriate one to classify the given input pattern. This solutions avoids the communication of large blocks of data and does not require data sharing, thus respecting privacy concerns. However, to our understanding, their mechanism of negotiation, which is asynchronous just to some extent, may be highly affected by the order with which agents try to negotiate and choose their negotiation partners. Therefore, when selecting the agent with highest confidence to provide the consensus classification for an unlabeled instance, there is no guarantee that its model fully incorporates the knowledge derived by other agents. Conversely, in the approach herein proposed, every agent necessarily takes part in the final classification decision.

JAM [35] is a non-collaborative agent-based system that proposes an unifying and scalable solution based on meta-learning to treat data mining tasks for large and potentially distributed data sets. One of JAM's primary uses is fraud detection in banking domains. The JAM framework provides a collection of classification algorithms to be encapsulated in the learning agents and a meta-learning agent that integrates multiple models, possibly learned at different sites, into a global model. Hence, JAM supports DDM with heterogeneous data sources. Nonetheless, the combination of the mod-

els induced locally by each learning agent based on their accessible features is performed by means of voting or meta-learning techniques (e.g., arbitration rule), which according to Tumer and Ghosh [38] are unsuitable techniques to deal with heterogeneous environments.

Similarly to JAM, Papyrus [3] is a Java-based system that addresses DDM over clusters of heterogeneous data sites and meta-clusters. Each cluster has one distinguished node that acts as the control point for the agents. Papyrus is designed to support different tasks, algorithms and model combination strategies. Some combination methods implemented by Papyrus are simple voting, meta-learning techniques and knowledge probing [14]. Differently from previous tools that avoid data sharing, Papyrus support explicit data transfer among sites or clusters. For instance, agents are able to move data, intermediate results or predicted models either among sites, or from local sites to a central root, in order to produce the final result. Therefore, Papyrus is not suitable to applications that require data privacy. Nonetheless, for overall DDM problems in which transferring data is acceptable and feasible, Papyrus seems a promising approach given the flexibility of the framework to a wide range of applications.

Kargupta and colleagues developed the BODHI (Besizing knOwledge through Distributed Heterogeneous Induction) system [20], a non-collaborative agent-based system designed for collective data mining tasks in heterogeneous environments. The difference of this system in contrast to previous approaches is that BODHI aims at finding globally meaningful pieces of information from each local site and use it to build the global model, instead of combining incomplete local models. The main idea is that any function can be represented in a distributed manner using an appropriate set of basis functions. Therefore, data modeling is not performed using popular representations such as decision trees, neural networks, among others. Instead, BODHI learns the spectrum of these models using an orthonormal basis space and then converts the model from orthonormal representation to a popular form. This approach guarantees correct local and global models without requiring high data communication loads. The initialization and coordination of the system is performed by a central facilitator, which is also in charge of regulating communication and control flow between agents.

Although many DDM systems adopt a multi-agent system architecture, other interesting works offering alternative approaches to DDM problems with vertically partitioned data are also found in literature. For instance, Tumer and Ghosh [38] propose an ensemble approach to combine local classifiers inferred from multiple heterogeneous data sites. Their solution uses an order statistics-based technique that first orders the predictions of different classifiers for each possible class and then aggregates the orderings in an appropriate manner to generate a consensus solution. A combination strategy proposed by the authors consist in selecting the kth ranked output for each class as representing its posterior, in which kth is defined in terms of

the maximum, median or minimum value of the ordering. Moreover, authors also suggest the combination of averaging and order statistics to derive linear combiners for pooling classifier outputs. More precisely, they propose to average the maximum and minimum values (the "spread" combiner) and to define the posterior estimate around the median values of multiple classifiers (the "trimmer" combiner).

Another alternative approach for dealing with vertical data partitions, which also does not implement an agent-based data mining system, was proposed by Matatov and colleagues [25]. In their work, authors use naïve Bayes combination to merge the predictions of several models and classify unlabeled instances. However, their motivation for dealing with this type of DDM problem is different from the aforementioned works. While in previous efforts vertically distributed data was an inherent feature of the target domain, Matatov and colleagues create a distributed environment as a solution to deal with data privacy concerns, yielding the so-called "data mining privacy by decomposition" (DMPD) method, which considers anonymization for classification through feature set partitioning. Specifically, they adopt a k-anonymity framework for data mining privacy as proposed by Sweeny [37] and use a genetic algorithm to search for optimal feature set partitioning. differences among this work and previous ones, authors explore an interesting new combination method that has not been used by agent-based approaches.

Finally, we emphasize that there are others well-known agent-based systems designed for DDM problems, such as PADMA [19], KDEC [22] an the multi-agent system proposed by Agogino and Tumer [1]. These works have in common the fact that the privacy of data is preserved since solely high-level information derived from data analysis is shared among agents. Nonetheless, they focus on distributed clustering tasks and assume horizontally partitioned data (homogenous environments), characteristics that diverge from the research interests of the present paper.

3 Methods

As previously noted, in some real-world applications data centralization is neither desirable or possible due to concerns regarding data privacy and confidentiality, or even as a consequence of bandwidth limitation and storage constraints. This obviously poses more challenges to the data mining task, since most of the existing data mining techniques have been designed assuming centralized and fully accessible training data [22].

Here, we are specifically interested in distributed classification tasks with vertically partitioned data, i.e., when features are distributed among heterogeneous data sites. As general DDM problems, distributed classification requires distributed data analysis and generation of local models based on

agents' accessible knowledge. To this end, each agent has the personal goal of deriving the most accurate possible model to classify new instances on the basis of their training set containing instances whose class is known, similarly to conventional classification tasks. Nonetheless, because local models are at best incomplete, at a second stage agents need to communicate their findings either to other agents or to a facilitator in order to enable the generation of a global and more accurate model. It is our concern to accomplish this step preserving existing data privacy.

In what follows we define the methods embedded in the two entitiesthat compose the architecture of the proposed agent-based DDM system: the learning agents, responsible for the induction of the local models, and the facilitator, whose task is to derive the global model.

3.1 Learning agents

Learning agents operate on local databases and are responsible for learning the relationships among the features values and the label (class) of each instance in the training set, thereby devising a model that can be later applied to predict the label of new instances. For binary classification tasks, which is the case of our example application (see Section 5), classes refer either to 'positive' or 'negative' value, being the positive class the one in which we are interested (the target class).

The input to learning agents takes the form of a training set of labeled instances E, each of which is identified by an unique *id*. Furthermore, each instance is described by a vector of features F, divided into not necessarily disjoint subsets $F_i \subseteq F$, with $i \in 1, \ldots, n$. Given a collection $\mathcal{A}$ of n agents and assuming vertically partitioned data, each agent a_i knows the correct class (label) of every training instance, but has access only to the subset of features F_i available at its local database. In other words, all agents are aware of properties such as the number of instances covered by the training set, their respective *ids* and labels, but each agent has a particular and partial knowledge regarding the features that describe these instances. As output of the training process, each agent returns the classifier model and a list of predicted class labels and class probabilities for each instance in the input training set, which may be used to assess their performance.

In the case of unlabelled instances, the input and output follow the same standard as in the training process, with some minor differences. As input, agents also receive a subset F_i of features describing instances that are identified by their unique *ids*, except that their true class labels are not available. In what concerns the output, agents return solely the class predictions, both the labels and probabilities.

The number of agents and the machine learning algorithms that they encapsulate are flexible properties of the system. Although all learning agents are built over the same set of methods (i.e., they all implement functions

that allow, for instance, their initialization, learning the model, or sharing their predictions with other agents), agents may rely on distinct machine learning algorithms to build their local models. In the present paper, we investigate the problem of combining multiple classification results in a distributed classification task by considering a scenario in which each agent embeds a distinct machine learning algorithm. This can be interpreted as an analogy for people with distinct expertise or background working together on the same task.

Following this direction, learning agents are implemented as R functions and encapsulate machine learning algorithms that are available in R packages such as `e1071` [27] and `RWeka` [16]. In the present paper, we devise a DDM system composed of five agents, each of which implements its methods for training a classifier upon one of the following algorithms:

- **JRip:** an implementation of a propositional rule learner, the Repeated Incremental Pruning to Produce Error Reduction (RIPPER)[8];
- **J48:** a Java implementation of the C4.5 algorithm for classification via decision trees [31];
- **KNN:** the K-nearest neighbor algorithm, an instance-based learning classifier [2];
- **NB:** naïve Bayes, a probabilistic classifier based on the Bayes theorem [18];
- **SVM:** support vector machine, a classifier based on the concept of hyperplanes [7].

Although only a subset of classifiers were used for test purposes, we remark that any classifier can be adopted by the proposed agent-based system given its availability in R packages. Moreover, since we are not concerned in optimizing the performance of a single classifier, and neither are interested in comparing the efficiency and predictive power of distinct classifiers, we applied the above classifiers using their standard parameters, which can be found in their description in the aforementioned R packages documentation.

Finally, since the classifiers enumerated above are not bootstrapping methods, in order to gather unbiased statistics regarding the classification results, training and testing data sets must be clearly specified. Therefore, the data set related to our application domain (see Section 5) was separated into two independent sets, training and testing set, based on a resampling method, allowing agents to build their classification model and assess their strength and utility as a predictive system. However, it is important to notice that in the experiments discussed in Section 6, all classifiers are trained upon the same random subsets of training instances and tested with the same random subsets of testing instances, thus allowing a fair comparison of their performance.

3.2 Facilitator agent

As has been noted, an intrinsic issue in DDM tasks is how to integrate the knowledge individually learned by a set of agents into a globally coherent model. This issue becomes even more challenging when agents' local models are learned from heterogeneous databases [28, 39]. Given that agents work at distributed data sites, each of which comprises a partition of the features space, none of the agents can individually learn the concepts with high accuracy level, since the success in classification tasks is closely related to the quality of the features vector F [11]. Combining these models is thus a non trivial task. Simple voting mechanisms and meta-learning approaches have been considered unsuitable to deal with this problem for a number of reasons, including the instability of the performance of locally built models and the impracticability of training a meta-classifer when massive data is being generated in geographically distributed sites [38].

In the present paper we address this issue by proposing new combination mechanisms inspired by the social choice theory that are more sophisticated than the simple majority voting, but do not compromise a good efficiency in what concerns the trade-off between communication and accuracy. Social choice functions deal with the problem of aggregating many individual preferences into a single collective preference [33, Chapter 9]. Here, we show that the application of this method is not only feasible, but also robust in the case of distributed classification, even upon vertically partitioned data.

In the scope of this work, individual preferences are defined in terms of agents' predictions, i.e., the predicted class labels and class probabilities for the given input data set. Therefore, once local models have been built and applied for the classification of the input data, learning agents may either share their preferences with each other or transfer them to a facilitator agent, which acts as a central facility in which the global model is assembled. For the sake of simplicity, here we assume the latter situation. Nonetheless, it is important to note that whenever this type of centralization is not tolerated, the same results can be achieved if agents broadcast their preferences and each agent runs a combination function locally.

To introduce the process of deriving the global model, let $\mathcal{A} = \{a_1, a_2, ..., a_n\}$ denote a set of n agents in our system, and let O denote a finite set of outcomes, which in our application refers to the instances comprised in the input data set. In a classification task, each agent produces as output predictions about the most probable class label and the estimated classes probabilities for each $o \in O$. Assuming a binary classification task in which labels may be either 'positive' or 'negative', a probability threshold of 0.5 is used to define the predicted class label: probabilities higher than 0.5 yield a prediction for the 'positive' class, while a probability equal or lower than 0.5 predicts for the 'negative' class. Based on the predicted class probabilities, agents define their preferences in relation to the instances that they consider more likely

to belong to the 'positive' class, which is our target class.

Because preferences are transitive, an agent's preference relation induces a linear ordering L_i, which is a total strict ordering on O. Thus, we adopt the usual notation and use $o_{j-1} \succ_i o_j$ to capture strict preference of agent a_i, i.e., agent a_i prefers outcome o_{j-1} to outcome o_j. In a classification context, this means that the probability attached by agent a_i concerning the target class is greater for outcome o_{j-1} than for outcome o_j. In this process, instances associated to high probabilities will be "preferred" over instances associated to low probabilities, thus yielding a structure similar to a ranking over the set of outcomes O. Given that all agents work on the same input data, their preferences ordering is defined over the same set of outcomes O, enabling a comparison and eventually an integration of this information. Nonetheless, because agents run different machine learning algorithms and build their models over distinct features subsets, the actual ordering of outcomes is very likely to differ among agents.

Once the classification is over at the learning agent level, each agent transfers its preference ordering L_i to the facilitator agent. As already mentioned, each instance in the input data is identified by an unique *id*, which is the only information from the original data set comprised in agents' preferences. Therefore, data communication in terms of agents' preferences guarantees that solely high-level information about agents' private knowledge will be shared among agents, which complies with the requirement of preserving data privacy and allows the overall agent-based system to pursue more satisfactory results in the DDM problem. Moreover, this approach has the advantage of notably reducing the load of data transferred among data sites, since communicating a linear ordering defined over the instances in the input data set is less expensive than communicating the complete data set, even when this alternative is acceptable.

When in possession of the learning agents' preferences, the facilitator agent defines the preference profile $[\succeq] \in L^n$ as a n-tuple containing the ordering on O provided by the n learning agents and apply over this tuple a social choice function (SCF). The goal of a SCF is to systematically transform individual preferences into a social decision, producing the final or consensus preference ordering that best reflects the preferences of all n agents regarding the classification of O. Therefore, the social choice can be interpreted as a mapping function $f : L^n \mapsto L$, where L is the group preference regarding the ordering over all O possible outcomes. Although in literature the term SCF is often used specifically for the case where a single outcome (candidate) is selected from a set of preferences, in the current work we use this term in a broad sense. From the group preference L, one can extract information such as the top-ranked outcome in terms of 'positive' class probability, or the set of outcomes that are more likely to belong to the 'positive' class, among others.

Here, we test three SCFs as combination methods in our agent-based

DDM system:

- **Borda count:** Borda's method works by assigning to each outcome $o \in O$ a score computed as a function of its position within each agent's ranked list of preferences. Therefore, for each outcome o and preference ordering L_i, $B_i(o)$ equals the number of candidates ranked below o in L_i (o inclusive). The total Borda score $B(o)$ for outcome o is then defined as $B(o) = \sum_{i=1}^{n} B_i(o)$, i.e., a sum over the scores computed by the n learning agents. The final step of Borda's method consists in devising an aggregated rank by averaging the Borda scores by the number of learning agents in the DDM system and the cardinality of O.

- **Copeland function:** in Copeland function [9], the score $C(o)$ of each outcome $o \in O$ is computed as the number of pairwise victories minus the number of pairwise losses in relation to every possible element of O. Wins and losses are defined in terms of the position that each outcome occupies in the agent's preferences: when comparing two candidates, the outcome that is preferred by the majority of learning agents, i.e., the one that has a higher probability for the target class, is said to be the winner of one-on-one contests. The social choice by Copeland function aims at identifying the outcomes with the greatest number of net wins. The philosophy under this method is that if a simple majority win is good for an outcome, then the more the better.

- **Footrule function:** the Footrule function is a good approximation of the Kemeny optimal aggregation and is related to the median of the values in a position vector. Given total preference orderings $L_1, \ldots, L_n$, if the median positions of the candidates in the lists form a permutation, then this permutation is a footrule optimal aggregation. It can be shown that a permutation minimizing the total footrule distance to the L_i's is given by a minimum cost perfect matching in a bipartite graph [12]. In this sense, the first set of nodes or outcomes O denotes the set of elements to be ranked, while the second set of nodes P, with cardinality equal to O, denotes the m available positions in the rank. The Footrule function looks for the permutation that minimizes the weight $W(o; p) = \sum_{i-1}^{m} |L_i(o) - p|$ of a ranking that places outcome o at position p.

We compare the above SCFs against the plurality voting, in which each agent casts a single vote regarding an outcome, or instance, and the predicted class of each outcome is defined as the one with most votes. For binary classification tasks, plurality voting replays a simple majority voting, which is a widely used aggregation method in problems of DDM, collaborative learning and ensemble learning [29, 1, 32, 10].

Besides our interest in addressing DDM problems with vertically partitioned data and data privacy concerns, the use of SCFs is also motivated by the so-called "wisdom of crowds" theory [36]. According to Surowiecki [36], collective decisions may provide more robust solutions in contrast to individuals' or even experts' decisions, given that agents in the crowd have different expertise or knowledge about the problem under consideration. The diversity raised by this scenario is often an advantageous feature in decision-making problems and has been proved to result in accuracy improvements in several practical applications in the fields of DDM and ensemble learning [35, 10].

4 Classification with the proposed agent-based DDM system

For the experiments run in this study, we devise an agent-based classification system that is heterogenous in two sense, i) each agent has access to a partial set of features and ii) each agent builds its local model using a different classifier from the list presented in the previous section. Nonetheless, we remark that the system is very flexible and the addition of more learning agents, as well as an agent-based DDM system with homogeneous agents, are straightforward modifications. In what follows, we discuss the functioning of our agent-based classification system (Fig. 2a).

The first step consists of the definition of the training data. In the distributed classification task, each learning agent a_i is assigned to a particular data site and access its private $F_i \subseteq F$ subset of features, such that each agent builds its classifier with a partial view of the problem. Once assigned to a data site, the subset of features available in that site is considered as a private information of the agent. However, as already mentioned, all agents share the *ids* of the training or unlabeled instances as a common information, such that at the end of the process agents can communicate to generate a consensus, global model.

Next, as in conventional classification tasks, a training process takes place, whose procedure is specific to the machine learning algorithms adopted. In this phase, learning agents induce the relationships between their features values and the expected class labels of instances in the training set using as basis their particular expertise, i.e., their embedded ML algorithm. In the scope of this work, the DDM system is composed of five agents who implement popular ML classifiers available in R packages: JRip, J48, KNN, NB and SVM. Because learning agents differ in terms of their private knowledge (features set) and expertise (ML algorithm), classifier models of distinct predictive power and generalization errors are expected, introducing diversity into the system.

The simplicity of SCFs eliminates the need to train the facilitator agent,

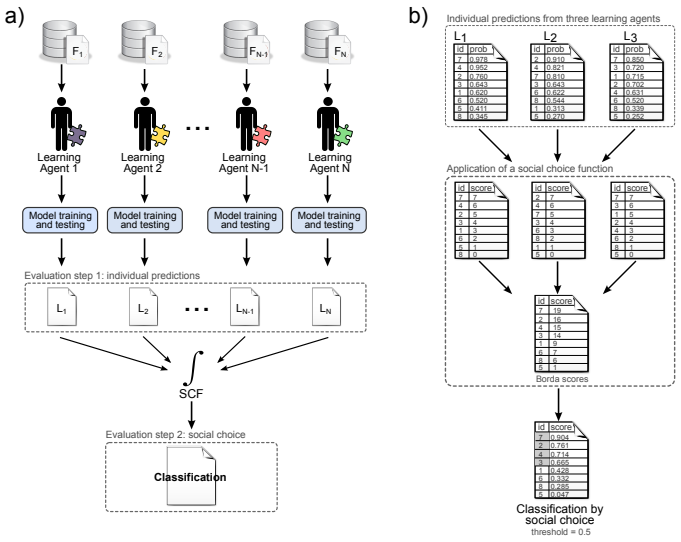


Figure 2: Framework of the proposed agent-based classification system. a) For distributed classification tasks, each learning agent has a partial view of the problem $(F_1, \ldots, F_N)$ and build its individual model based on its private features and expertise (embedded classifier). Individual predictions are them shared and combined into a consensus prediction based on the SCFs proposed. b) An example of distributed classification task among three agents shows how consensus is created by a SCF based on agents' preferences, given in terms of the probability for the target class. In this example, Borda count is used to aggregate preferences. The only information shared among agents is the instances' *ids*. The SCF outputs a consensus preference ordering, and instances are then classified based on a threshold (0.5 was used in this example, where the highlighted instances are assigned to the target class).

since there are no parameters to be optimized. Therefore, the training of learning agents' models is followed by a testing step, which aims at assessing the quality of learning agents' predictions and allow performance comparison between individual predictions and social prediction. As in the training process, the testing data is the same for all agents, i.e., it comprises the same instances, except that in a distributed classification scenario the data set is vertically partitioned among learning agents. At the end of the testing step, each learning agent generates their preferences regarding instances's probability for the target class. At this point, two processes take place: preferences are shared among agents or transferred to a central facility, and the quality of agents' individual predictions is assessed.

The evaluation is performed by means of a Receiver Operating Characteristic (ROC) curve, which is a plot that illustrates the performance of a

binary classifier system as the decision threshold varies. Thus, each point on the curve represents the true positive (TP) rate and false positive (FP) rate at a different decision threshold. Ideally, one expects the y-values (TP rate) to grow at a faster rate than their x-values (FP rate). Additionaly, we compute the area under the ROC curve (the AUC score). This score can be interpreted as the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. Thus, the higher the AUC score, the better the predictive accuracy of the classifier.

The final step of the system's functioning takes all preference orderings of the learning agents as input and applies one or more SCFs in order to find the social choice for the classification of each unlabeled or test instance, as exemplified in Fig. 2-b. This can be accomplished in two ways, as explained in Section 3.2, i.e., either by a facilitator agent or at each of the agents' site once they have all exchanged their preferences. Although the former option can be seen as a centralization point, we emphasize that there is no centralization nor disclosure of the original data used for classification. In Fig. 2, we illustrate the case where a facilitator is responsible for this task. Observe in this practical exemple that the learning agent's preferences contains solely the classifying instances identified by their unique *id*, ordered by the predicted probability for the target class. In this example, Borda count is applied by the facilitator agent, such that probabilities are converted into Borda scores and a pre-defined threshold (usually 0.5) is applied over the consensus preference ordering to obtain the final classification.

In summary, the steps performed by the proposed agent-based DDM system are as follows:

1. Learning agents are assigned to the distributed data sites, each of which contains a subset of the features space;

2. Learning agents train their classifier models using their embedded ML algorithm and their respective partition of the data;

3. Learning agents test their classifier models based on the same testing data set, following the vertical partition of data;

4. Individual classification results are assessed by means of ROC curves and AUC scores (for comparison purposes);

5. Learning agents either exchange their preferences or send them to a facilitator agent;

6. A SCF is applied, which outputs a consensus preference ordering that reflects the social choice;

7. The consensus prediction is assessed by means of a ROC curve and corresponding AUC score.

To assess the performance of the proposed DDM system, we repeat the process described through steps 1–7 several times, each of which with a distinct distribution of agents among the data sites such that the results are not biased towards any distribution of features. For comparison purposes, we also execute a conventional classification task, in which data is centralized and hence agents train their classifiers upon a complete view of the classification problem. We follow the same methodology represented by the framework in Fig. 2-a, with the difference that the data sets $F_1, F_2, \ldots, F_N$ are exactly the same. This scenario is very similar to the tasks tackled by ensemble learning methods, such that our system could also be used as an alternative solution when the interest relies in aggregating different classifiers in order to boost performance and mitigate the uncertainty about classifiers performance in unknown data sets.

5 Application domain

In the current paper we are interested in classification of biological data, which is a central task in the field of bioinformatics. This refers to a real-world domain in which distributed classification tasks may arise. First, algorithms may not be able to access all data due, for instance, to intellectual property rights and privacy issues. As an example, different laboratories working towards the same problem might be willing to collaborate but keeping full control of the data they generate. Second, biological data are often large and derive from decentralized sources (multiple databases), such that high availability of bandwidth or storage space would be required to centralize the data. In scenarios like this, in which it is not possible to assume that all training data is concentrated in a single source and available to agents at all times, one can formulate a distributed classification task by assigning each of the data subsets to a distinct learning agent. In what follows we briefly introduce the biological problem addressed in the current paper and explain the data used for classification.

5.1 Prediction of microRNAs target genes

The study of gene expression and regulation is one of the most explored problems in bioinformatics. Gene expression profiling for hundreds or thousands of genes are analysed in order to unveil the regulatory interactions among these genes. However, nowadays it is known that many genes' expression might be regulated by elements derived from noncoding RNA sequences, rather than protein coding sequences. Therefore, the identification of such noncoding RNA sequences and the genes they target has become an important challenge in the field. MicroRNAs (miRNAs) is a family of highly conserved small noncoding RNA molecules, measuring about 22 nucleotides long, that causes negative regulation of gene expression. This regulation

is post-transcriptional, i.e., occurs at the RNA level between the transcription and the translation of a gene, and consists of gene silencing by target degradation or translational repression [4].

In this sense, miRNAs have been reported to act in processes such as developmental timing, metabolism, differentiation, proliferation and cell death. Furthermore, miRNAs can act as both tumour-suppressor genes and oncogenes [23]. Thus, the determination of miRNAs targets is an increasingly active area of research. Among the most common approaches, ML methods have achieved the best results so far in identifying miRNAs target genes. Based on a set of descriptive features from the interaction of miRNAs and their true and pseudo targets, such as characteristics of bases complementarity, thermodynamic stability, and evolutionary conservation among species, these methods train a classifier to identify the class of new unlabelled instances. For a more detailed discussion about the biogenesis and regulatory mechanisms of miRNAs, we refer reader to the paper published by Bartel [4].

5.2 Features

In the current paper, we use the data applied by Mendoza and colleagues [26] in the training process of a random forest classifier for human miRNA target genes prediction to train the proposed agent-based classification system. This data set is composed by 478 positive instances of miRNA-target pairs and 286 negative ones, each of which described by a set of 34 features.

We divide these features into five semantic groups: structural, thermodynamic, alignment, position-based and seed-related features. The seed region is a binding region between miRNA and its target composed in general by 8 nucleotides that is known to play a crucial role in the identification of miRNAs targets [4]. Therefore, the five features groups used in this study (and their corresponding set dimension) are defined as follows:

- **Structural features (5 features):** number of Watson-Crick matches (G:C and A:U pairing) and mismatches (G:U wobble pair, gap and other mismatches) in the miRNA-target alignment, given by integer values ranging from 0 to 20.

- **Thermodynamic feature (1):** minimum free energy (MFE) of the alignment between a miRNA and its target computed by the RNAfold program from Vienna package [15], given by negative values ranging from -41 to -10.

- **Alignment features (2):** score and length of the miRNA-target alignment, computed by the miRanda software [13], given by integer values ranging from 140 to 181 (alignment score) and from 7 to 26 (alignment length).

- **Position-based features (20):** nominal values to designate the kind of nucleotide pairing in each position of the alignment, from the 5'-most position of the miRNA up to the 20th position: a G:C match, an A:U match, a G:U match, a gap and a mismatch.

- **Seed features (6):** thermodynamic and structural features concerning nucleotides 2–8 (seed region).

Each of these categories provide a different perspective, or evidence, for the study of miRNAs' candidate targets. Therefore, it is reasonable to think that this type of data may be generated in distinct data sites across multiple research groups, which are willing to collaborate in order to enhance the overall competence regarding the identification of new genes whose expression is repressed by miRNAs. As we have discussed, many situations in which collaboration is desired are impaired by the fact that data is under privacy or intellectual property concerns, and thus it can not be explicitly shared or transferred. Here, we assume this scenario and address the question whether prediction of miRNAs target genes can be effectively and efficiently accomplished under vertically distributed data. For further details about the biological data used in the current paper, we refer reader to the work of Mendoza et al. [26].

6 Results

To test the effectiveness of the proposed approach, we applied the agent-based classification system described in Sections 3 and 4 to the problem of identifying microRNAs target genes (see Section 5). Biological data sets have a great propensity to be physically distributed and, in many cases, can not be shared due to intellectual property rights. Still, this is an area where people are often willing to collaborate in order to catalyze scientific progress, but with restraint regarding data control. Therefore, it is a very motivating domain for our agent-based classification system.

As depicted in Fig. 2, the agent-based classification system comprises two evaluation points: the first one assesses the performance for learning agents' models, while the second appraises the predictions made upon the social choice. For both of them, we consider the case of centralized data (agents have a complete view of the problem) and distributed data (agents have a partial view of the problem, i.e., a subset of features).

As we are dealing with variation in classifiers and in data, we evaluate the performance of our method based on a 10-fold cross-validation in order to prevent overfitting. In each fold, we run all the steps of our system, i.e., training, testing and application of SCFs, performing a random distribution of learning agents among the data sites for the case of distributed data. This means that in each fold, agents receive a randomly assigned features

group (as defined in Section 5), based on which its model is built. Thus, the number of features accessible by each agent varies throughout the multiple folds according to the assigned features group. At the end of cross-validation, we average results over all runs to generate a performance measure for each agent. For the ROC curves, the average is performed taking vertical samples of the ROC curves for fixed FP rates and averaging the corresponding TP rates. The average AUC scores are computed as the mean among all AUC scores over the 10-fold cross-validation.

The main results are summarized in Table 1. The left side of the table shows the evaluation of the local models produced by the five learning agents, whereas the right side of the table depicts the results of the global models based on the social choice – plurality (PLU), Borda count (BOR), Copeland's Function (COP) and Footrule function (FOO).

Compared to the centralized scenario, a large variance is observed for the performance of the classifiers when data is distributed. This is due to the fact that distinct runs of the classification are performed over a distinct subset of features, and illustrates the explicit or implicit bias of algorithms that causes them to hold a preference for certain generalizations over others. In fact, agent 3, which runs a K-nearest neighbor classifier (KNN), has a poor generalization over the complete set of features, but achieves a superior generalization for some of the features groups, explaining its higher mean and variance for the distributed case. Thus, in situations where the system is composed by heterogeneous agents, the approach of selecting a single agent to produce the output, as suggested in previous works [29, 32], is not the most appropriate.

Comparing the classification task using centralized and distributed data, the fact that in the latter agents build their individual models based on a vertical partition of data in general impairs the classification performance, as expected. Not only the means are lower, but the deviations are higher. One can be observed that among the learning agents, the best performance is associated with agent 1, which encapsulates a propositional rule learner (JRip) and achieves an average AUC score of 0.681.

Regarding the approach based on SCF, the average AUC scores in Table 1 corroborates the effectiveness of our approach for tackling the problem of distributed classification tasks. One can observe that computing the social choice of a set of agents can in fact help counterbalance the shortage of information generated by distributed data, as well as provide predictions as accurate as the ones produced by the agents. Whereas for complete knowledge there is a slight improvement of the ensemble system over the best individual learner, namely a difference equal to 0.032 between Copeland's and J48's average AUC scores, for partial knowledge the difference between the best ensemble (i.e., Copeland function) and the best individual (i.e., JRip) predictions raises to 0.1404 – a four-fold increase among these values.

Table 1: Average AUC scores of our agent-based classifier system computed after 10-fold cross-validation. We highlight the best scores for local models and global models for each of the scenarios (centralized and distributed data).

		Learning agents (local models)					**Social choice** (global models)			
		Agent 1	Agent 2	Agent 3	Agent 4	Agent 5				
		JRip	J48	KNN	SVM	NB	PLU	BOR	COP	FOO
Centralized data	Mean	0.799	**0.812**	0.580	0.716	0.709	0.788	0.821	**0.844**	0.796
	SD	0.065	0.041	0.069	0.046	0.061	0.049	0.046	0.030	0.053
Distributed data	Mean	**0.681**	0.573	0.668	0.635	0.595	0.770	0.742	**0.814**	0.720
	SD	0.143	0.131	0.172	0.163	0.104	0.106	0.096	0.080	0.081

Moreover, Borda count and Copeland function have performed specially well, achieving AUC scores of 0.740 and 0.814, respectively, and outperforming agent 1, which presents the best individual performance. The superior performance of the global models was also perceived for the conventional classification task. The use of SCFs improved classification results while keeping the standard deviation values relatively low. Again, the social decision provided by Copeland and Borda functions have depicted the best performance among the tested methods.

The ROC curves corresponding to local models (learning agents) and global models (SCFs) are depicted in Fig. 3 and Fig. 4 for centralized and distributed data, respectively. These plots show the average ROC curves overlaid by boxplots for fixed values of false positives (x axis). The box plots denote the median, maximum and minimum values, as well as the upper and lower quartiles. The dots outside the box represent (suspected) outliers, while the filled dots are the average values used to plot the average ROC curve.

Analysis of these results clarify even more the higher variance and inferior performance of the classification drawn by the individual agents in relation to the social choice among agents. In addition, readers can notice that the Copeland function produces the best performance in terms of the ROC curve, and also has the lowest variance in performance, together with the Footrule function. Moreover, when comparing the four aggregation methods used, plurality presents the highest variances and smallest lower quartiles among all methods, specially for low rates of false positives. The good performance of Copeland and Footrule functions comes at a cost, though, since they are more computationally demanding than Borda count and plurality voting.

Finally, we compare the AUC scores computed by the local models and the global models as assessed by a 10-fold cross validation in terms of their density probability distributions. As shown in Figure 5, the social choice (global model) tends to generate classifiers with higher predictive accuracy, indicated by the density distributions shifted to the right in relation to the local models. This is specially true for the case of vertically partitioned data (Figure 5-b). We perform a Mann-Whitney test and find statistically significant differences (p-value < 0.01) between the density distributions of global models in contrast to local models.

Although there are more scenarios and data sets that could be explored, preliminary results presented in this work suggest that more sophisticated SCFs have greater predictive accuracy than elemental voting procedures such as plurality voting. In addition, they present a good balance between cost of the computation and accuracy, thus representing a promising solution to deal with the problem of distributed classification tasks.

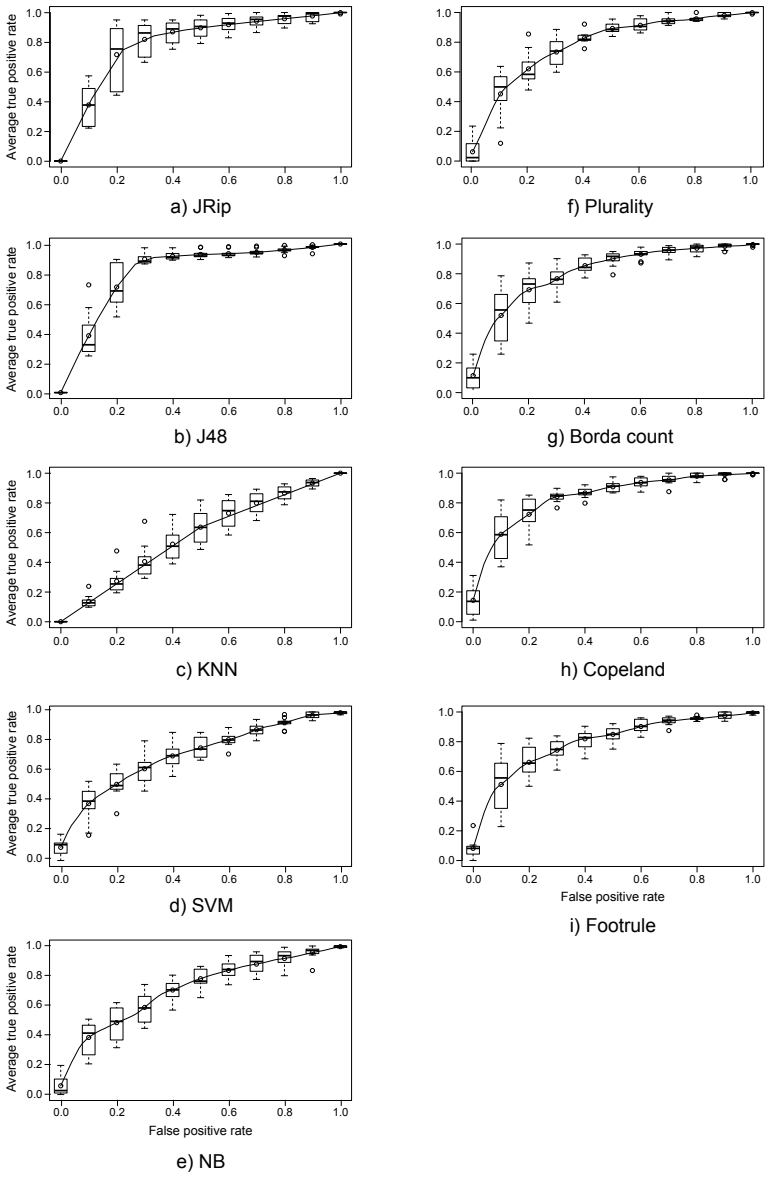


Figure 3: ROC curves for the performance of individual agents (a - e) and of the social choice (f - h) in a conventional classification task (data is centralized and fully accessible by agents).

7 Conclusion

Distributed classification tasks are inherently more complex than conventional ones in that agents have a local, partial view of the training experience and thus, no agent can individually learn a concept with high accuracy levels. Therefore, communication, collaboration and competence sharing are

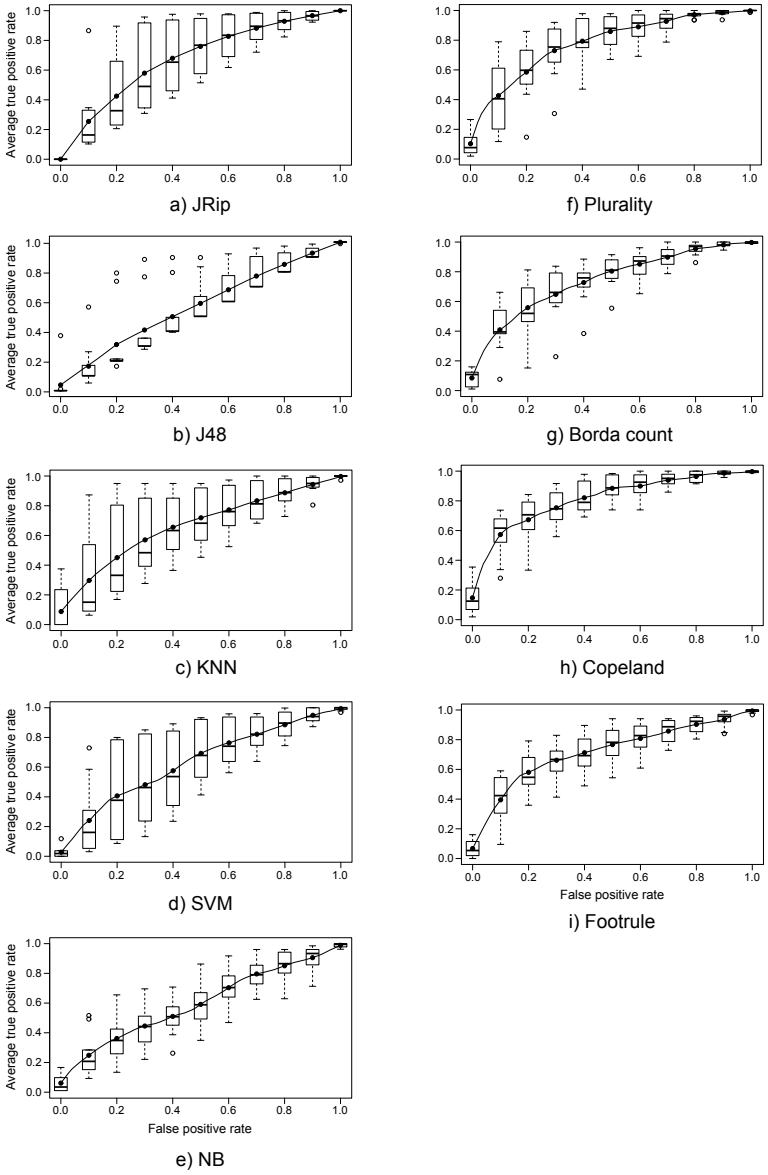
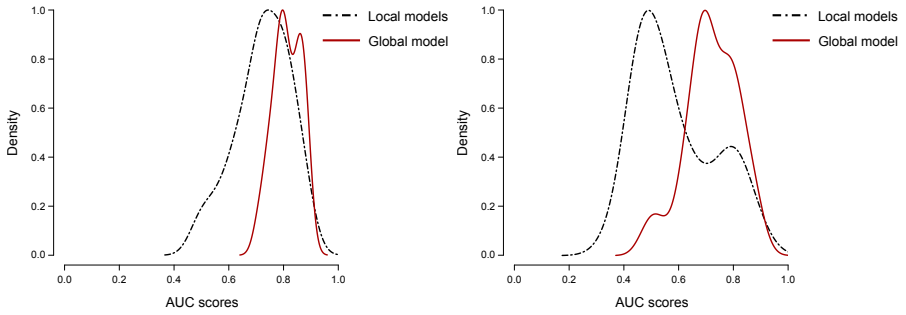


Figure 4: ROC curves for the performance of individual agents (a - e) and of the social choice (f - h) in a distributed classification task.

needed to build an accurate predictive model. This paper has proposed an agent-based classification system that aims at building a consensus domain among agents by means of social choice functions.

Particularly, we have tested and compared Borda count, Copeland function and Footrule function for a specific biological scenario in which distributed data might pose challenges to conventional classification algorithms, i.e., prediction of miRNAs target genes. Empirical evaluation suggests that



(a) Centralized data (b) Distributed data

Figure 5: Density distributions of the AUC scores produced by local models and global models for a 10-fold cross validation in the case of (a) centralized data and (b) distributed data. The right shift of the global models' distribution indicates the higher accuracy obtained by the SCFs, which is statistically significant in contrast to local models (p-value < 0.01, Mann-Whitney test).

the aforementioned social choice functions outperform the well known plurality voting method. Additionally, it was observed that social choice predictions are more reliable than individual predictions in two senses: they have higher average AUC scores and smaller variance across multiple classification runs. The superior performance of the prediction built upon the group decision was observed for centralized data as well, corroborating the fact that the combination of diverse learning algorithms with distinct classification bias is usually profitable and yields more robust classifying systems. Therefore, the devised system could also be used to solve conventional classification problems based on the ensemble learning paradigm.

Two important things to note about the system is that, first, formulation of the framework as described in the present paper aims at binary classification tasks. In this case, as mentioned in Section 3.2, preference orderings are defined according to the class probabilities computed by agents to input instances regarding the probability of belonging to a target class. For non-binary classification tasks, the agent's preference ordering can be formulated in terms of the probability attached to (a subset of) possible classes. Second, since our agent-based system performs aggregation over preference orderings, classification of unlabelled instances is performed in batch. For many applications, such as the one used as example in the current paper, this is a reasonable approach as one usually deals with very large data sets, from which we aim at extracting specific knowledge or refining the data set for more evident hypothesis that can be further confirmed by experimental validation.

For future work, it would be interesting to apply the agent-based DDM system to different problems, performing a broader and more thorough anal-

ysis of its performance across several scenarios. Also, one could expand this study to other social choice functions, as well as assess how their performance varies as function of properties such as size and homogeneity/heterogeneity of the agent-based DDM system.

8 Acknowledgments

This research and both authors are partially supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

References

[1] A. Agogino and K. Tumer. Efficient agent-based cluster ensembles. In Peter Stone and Gerhard Weiss, editors, *AAMAS '06: Proceedings of the 5th International Joint Conference on Autonomous agents and Multiagent Systems*, pages 1079–1086, New York, NY, USA, 2006. ACM.

[2] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[3] S. Bailey, R. Grossman, H. Sivakumar, and A. Turinsky. Papyrus: A system for data mining over local and wide area clusters and superclusters. In *Proceedings of the 1999 ACM/IEEE Conference on Supercomputing*, Supercomputing '99, page 63, New York, NY, USA, 1999. ACM.

[4] David P. Bartel. MicroRNAs: Genomics, review biogenesis, mechanism, and function. *Cell*, 116:281–297, 2004.

[5] Ana L. C. Bazzan. Agents and data mining in bioinformatics: Joining data gathering and automatic annotation with classification and distributed clustering. In Longbing Cao, editor, *Proc. of the Workshop on Agents and Data Mining Interaction*, number 5680 in Lecture Notes in Artificial Intelligence, pages 3–20, Berlin, 2009. Springer-Verlag.

[6] Longbing Cao, Gerhard Weiss, and Philip S. Yu. A brief introduction to agent mining. *Autonomous Agents and Multi-Agent Systems*, 25(3):419–424, 2012.

[7] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[8] W. W. Cohen. Fast effective rule induction. In *Proc. of the 12th International Machine Learning Conference*, pages 115–123, San Francisco, CA, 1995. Morgan Kaufman.

[9] A. Copeland. A "reasonable" social welfare function. Master's thesis, Seminar on Mathematics in Social Sciences, University of Michigan, 1951.

[10] Thomas G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15, London, UK, UK, 2000. Springer-Verlag.

[11] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, October 2012.

[12] Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 613–622, New York, NY, USA, 2001. ACM.

[13] Anton Enright, Bino John, Ulrike Gaul, Thomas Tuschl, Chris Sander, and Debora Marks. MicroRNA targets in Drosophila. *Genome Biology*, 5(1):R1, 2003.

[14] Yike Guo, Stefan M. Rüger, Janjao Sutiwaraphun, and Jodie Forbes-millott. Meta-learning for parallel data mining. In *Proceedings of the Seventh Parallel Computing Workshop*, pages 1–2, 1997.

[15] I. L Hofacker. Vienna RNA secondary structure server. *Nucleic Acids Research*, 31(1):3429–3431, 2003.

[16] Kurt Hornik, Christian Buchta, and Achim Zeileis. Open-source machine learning: R meets Weka. *Computational Statistics*, 24(2):225–232, 2009.

[17] A.K. Jain, R. P W Duin, and Jianchang Mao. Statistical pattern recognition: a review. *EEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[18] George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo, 1995. Morgan Kaufmann.

[19] Hillol Kargupta, Ilker Hamzaoglu, Brian Stafford, and Brian Stafford. Scalable, distributed data mining using an agent gased architecture. In *Proceedings of the Third International Conference on the Knowledge Discovery and Data Mining*, pages 211–214, Menlo Park, California, USA, 1997. AAAI Press.

[20] Hillol Kargupta, B Park, Daryl Hershberger, and Erik Johnson. Collective data mining: a new perspective toward distributed data mining. In *Advances in Distributed and Parallel Knowledge Discovery*, volume 2, pages 131–174. MIT Press, Cambridge, MA, USA, 1999.

[21] Hillol Kargupta, B Park, E Johnson, E Sanseverino, L Silvestre, and Daryl Hershberger. Collective data mining from distributed vertically partitioned feature space. In *Proceedings of the Workshop on Distributed Data Mining, International Conference on Knowledge Discovery and Data Mining*, pages 70–91. AAAI Press, 1998.

[22] Matthias Klusch, Stefano Lodi, and Gianluca Moro. Agent-based distributed data mining: The KDEC scheme. In Matthias Klusch, Sonia Bergamaschi, Pete Edwards, and Paolo Petta, editors, *Intelligent Information Agents*, volume 2586 of *Lecture Notes in Computer Science*, pages 104–122. Springer Berlin Heidelberg, 2003.

[23] Jeffrey Liu, Min Zheng, Ya ling Tang, Xin hua Liang, and Qin Yang. microRNAs, an active and versatile group in cancers. *Int J Oral Sci*, 3:165–175, 2011.

[24] Tinghuai Ma, Wenjie Liu, Yawei Zhao, and Zhong Liu. Sharing weather data in an agent-based distributed platform. In *Proceedings of the International Symposium on Computational Intelligence and Design (ISCID '08)*, volume 1, pages 541–544. IEEE Computer Society, 2008.

[25] Nissim Matatov, Lior Rokach, and Oded Maimon. Privacy-preserving data mining: A feature set partitioning approach. *Information Sciences*, 180(14):2696–2720, July 2010.

[26] Mariana R. Mendoza, Guilherme C. da Fonseca, Guilherme Loss-Morais, Ronnie Alves, Rogerio Margis, and Ana L. C. Bazzan. RFMirTarget: Predicting Human MicroRNA Target Genes with a Random Forest Classifier. *PLoS ONE*, 8(7):e70153, 07 2013.

[27] D. Meyer. *Support Vector Machines: The interface to libsvm in Package e1071.* Techische Universitat Wien, Austria, 2004.

[28] Pragnesh Jay Modi and Peter Woo Tae Kim. Classification of examples by multiple agents with private features. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 223–0229. IEEE Computer Society, September 2005.

[29] Pragnesh Jay Modi and Wei-Min Shen. Collaborative Multiagent Learning for Classification Tasks. In *Proceedings of the 5th International Conference on Autonomous Agents*, AGENTS '01, pages 37–38, New York, NY, USA, 2001. ACM.

[30] Andreas Prodromidis, Philip Chan, and Salvatore Stolfo. Meta-learning in distributed data mining systems: Issues and approaches. In *Advances in Distributed and Parallel Knowledge Discovery*, volume 3, pages 81–114. MIT/AAAI Press, Cambridge, MA, USA, 2000.

[31] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[32] Laura Emmanuella O. Santana, Anne M. P. Canuto, and Marjory C. C. Abreu. Analyzing the performance of an agent-based neural system for classification tasks using data distribution among the agents. In *IJCNN*, pages 2951–2958, 2006.

[33] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

[34] Josenildo C. da Silva, Chris Giannella, Ruchita Bhargava, Hillol Kargupta, and Matthias Klusch. Distributed data mining and agents. *Engineering Applications of Artificial Intelligence*, 18(7):791–807, October 2005.

[35] Salvatore Stolfo, Andreas L. Prodromidis Shelley Tselepis, Andreas L. Prodromidis, Shelley Tselepis, Wenke Lee, Dave W. Fan, and Philip K. Chan. JAM: Java agents for meta-learning over distributed databases. In *Proceedings of 3rd International Confence on Knowledge Discovery and Data Mining*, pages 74–81. AAAI Press, 1997.

[36] James Surowiecki. *The Wisdom of Crowds*. Anchor Books, a division of Random House, Inc., New York, NY, 2005.

[37] Latanya Sweeney. K-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, October 2002.

[38] Kagan Tumer and Joydeep Ghosh. Robust combining of disparate classifiers through order statistics. *Pattern Analysis and Applications*, 5:189–200, 2002.

[39] Li Zeng, Ling Li, Lian Duan, Kevin Lü, Zhongzhi Shi, Maoguang Wang, Wenjuan Wu, and Ping Luo. Distributed data mining: a survey. *Information Technology and Management*, 13(4):403–409, 2012.

[40] Haipeng Zheng, Sanjeev R. Kulkarni, and H. Vincent Poor. Cooperative training for attribute-distributed data: Trade-off between data transmission and performance. In *Proceedings of the 12th International Conference on Information Fusion*, pages 664–671, Seattle, WA, USA, 2009. IEEE Computer Society.