Data Min Knowl Disc (2012) 25:378–413
DOI 10.1007/s10618-012-0278-6

Network regression with predictive clustering trees

Daniela Stojanova · Michelangelo Ceci · Annalisa Appice · Sašo Džeroski

Received: 17 November 2011 / Accepted: 5 June 2012 / Published online: 22 June 2012

Abstract Network data describe entities represented by nodes, which may be connected with (related to) each other by edges. Many network datasets are characterized by a form of autocorrelation, where the value of a variable at a given node depends on the values of variables at the nodes it is connected with. This phenomenon is a direct violation of the assumption that data are independently and identically distributed. At the same time, it offers an unique opportunity to improve the performance of predictive models on network data, as inferences about one entity can be used to improve inferences about related entities. Regression inference in network data is a challenging task. While many approaches for network classification exist, there are very few approaches for network regression. In this paper, we propose a data

Responsible editor: Dimitrios Gunopulos, Donato Malerba and Michalis Vazirgiannis.

D. Stojanova (✉)
Department of Knowledge Technologies, Jožef Stefan Institute,
Jožef Stefan International Postgraduate School, Jamova cesta 39, 1000 Ljubljana, Slovenia
e-mail: daniela.stojanova@ijs.si

M. Ceci · A. Appice
Dipartimento di Informatica, Università degli Studi di Bari "Aldo Moro", via Orabona 4, 70125 Bari, Italy

M. Ceci
e-mail: ceci@di.uniba.it

A. Appice
e-mail: appice@di.uniba.it

S. Džeroski
Department of Knowledge Technologies, Jožef Stefan Institute, Jožef Stefan International Postgraduate School, Centre of Excellence for Integrated Approaches in Chemistry and Biology of Proteins, Jamova cesta 39, 1000 Ljubljana, Slovenia
e-mail: saso.dzeroski@ijs.si

mining algorithm, called NCLUS, that explicitly considers autocorrelation when building regression models from network data. The algorithm is based on the concept of predictive clustering trees (PCTs) that can be used for clustering, prediction and multi-target prediction, including multi-target regression and multi-target classification. We evaluate our approach on several real world problems of network regression, coming from the areas of social and spatial networks. Empirical results show that our algorithm performs better than PCTs learned by completely disregarding network information, as well as PCTs that are tailored for spatial data, but do not take autocorrelation into account, and a variety of other existing approaches.

Keywords Autocorrelation · Predictive clustering trees · Regression inference · Network data

1 Introduction

Networks have become ubiquitous in several social, economical and scientific fields, ranging from the Internet to social sciences, and including biology, epidemiology, geography, finance, and many others. Indeed, researchers in these fields have proven that systems of different nature can be represented as networks (Newman and Watts 2006). For instance, the Web can be considered as a network of web-pages, which may be connected with each other by edges representing various explicit relations, such as hyperlinks. Social networks can be seen as groups of members that can be connected by friendship relations or can follow other members because they are interested in similar topics of interests. Metabolic networks can provide insight about genes and their possible relations of co-regulation based on similarities in their expressions level. Finally, in epidemiology, networks can represent the spread of diseases and infections.

Regardless of where we encounter them, networks consist of entities (nodes), which may be connected to each other by edges. The nodes in a networks are generally of the same type and the edges between nodes express various explicit relations. Information on the nodes is provided as a set of properties (attributes), whose values are associated to each node in the network. The edges reflect the relation or dependence between the properties of the nodes. This is typically referred to as autocorrelation, that is, a cross-correlation of an attribute with itself (Cressie 1993).

In the literature, different definitions of autocorrelation are in use, depending on the field of study being considered. Not all of them are equivalent. In statistics, autocorrelation is generically defined as the cross-correlation between the attribute of a process at different points in time (Epperson 2000). In time-series analysis, temporal autocorrelation is defined as the correlation among timestamped values due to their relative proximity in time (Epperson 2000).

In spatial data analysis, spatial autocorrelation is defined as the cross-correlation of a property, which is measured across space. Spatial autocorrelation exists when there is systematic spatial variation in the values of a given property. This variation can exist in two forms, called positive and negative spatial autocorrelation. In the positive case, the value of a property at a given location tends to be similar to the values of that property in nearby locations. This means that if the value of some property is low at a

given location, the presence of spatial autocorrelation indicates that nearby values are also low. Conversely, negative spatial autocorrelation is characterized by dissimilar values at nearby locations. Positive autocorrelation is seen much more frequently in practice than negative autocorrelation. The main focus of analysis is positive spatial autocorrelation. This is also justified by Tobler's first law of geography according to which "everything is related to everything else, but near things are more related than distant things" (Legendre 1993).

In social analysis, autocorrelation can be recognized in the homophily principle, that is, the tendency of nodes with similar values to be linked with each other (McPherson et al. 2001). Homophily is observable, for example, in social networks where it is defined as the tendency of individuals to associate and bond with similar others (*friendship*). Actually, homophily shows that people's social networks are homogeneous with regard to many sociodemographic, behavioral, and intra-personal characteristics. Homophily effects among *friends* have demonstrated their importance in marketing (Chuhay 2010). Moreover, homophily is the hidden assumption of recommender systems, although it veers away from how people are socially connected to how they are measurably similar to each other.

In the context of *Twitter*, a special type of a social network, homophily implies that a *twitterer* follows a *friend* because he/she is interested in some topics the *friend* is publishing and the *friend* follows back the *twitterer* if he/she shares similar topics of interest. This is due to the fact that "a contact between similar people occurs at a higher rate than among dissimilar people" (Weng et al. 2010). Recently, Kwak et al. (2010) investigated homophily in two contexts: geographic location and popularity. They considered the time zone of a user as an approximate indicator for the location of the user and the number of followers as a measure for user's popularity. Among reciprocated users, they observed some level of homophily.

In this paper, we propose to mine regression models from network data by taking positive autocorrelation into account. The extracted models are tree-structured and represent reasonable clusters of training data. In this way, we combine descriptive and predictive mining in the same task. In the proposed approach, different effects of autocorrelation can be identified and considered at each node of the tree.

The paper is organized as follows. The next section clarifies the motivation and the actual contribution of this paper. Section 3 reports relevant related work. Section 4 describes the proposed approach. Section 5 describes the datasets, the experimental setup and reports relevant results. Finally, in Section 6 some conclusions are drawn and some future work outlined.

2 Motivation and contributions

The major difficulty implied by the presence of autocorrelation is that the independence assumption (i.i.d.), which typically underlies data mining methods and multivariate statistics, is no longer valid. The violation of the instance independence assumption has been identified as one of the main reasons responsible for the poor performance of traditional data mining methods (Neville et al. 2004; LeSage and Pace 2001). To remedy the negative effects of the violation of the independence

assumption, autocorrelation has to be explicitly accommodated in the learned models. In particular, recent research has explored the use of collective inference techniques to exploit this phenomenon when learning predictive models. According to Sen et al. (2008), collective inference refers to the combined classification of a set of nodes using the attribute value of a node and the labels of interlinked nodes. This means that, differently from traditional algorithms that make predictions for data instances individually, without regard to the relationships or statistical dependencies among instances, collective inference techniques collectively predict the labels of nodes simultaneously using similarities that appear among groups of interlinked nodes.

However, one limitation of most of the models that represent and reason for autocorrelation is that the methods assume that autocorrelation dependencies are stationary (i.e., do not change) throughout the network (Angin and Neville 2008). This means that possible significant variabilities in autocorrelation dependencies throughout the network cannot be represented and modeled. The variability could be caused by a different underlying latent structure of the network that varies among its portions in terms of properties of nodes or associations between them. For example, different research communities may have different levels of cohesiveness and thus cite papers on other topics with varying degrees. As pointed out by Angin and Neville (2008), when autocorrelation varies significantly throughout a network, it may be more accurate to model the dependencies locally rather than globally.

In this work, we develop an approach to modeling non-stationary autocorrelation in network data by using predictive clustering (Blockeel et al. 1998). Predictive clustering combines elements from both prediction and clustering. As in clustering, clusters of examples that are similar to each other are identified, but a cluster description and a predictive model is associated to each cluster. New instances are assigned to clusters based on the cluster descriptions. The associated predictive models provide predictions for the target property. Predictive clustering is similar to conceptual clustering (Michalski and Stepp 1983) since, besides the clusters themselves, it also provides symbolic descriptions (in the form of conjunctions of conditions) of the constructed clusters. However, in contrast to conceptual clustering, predictive clustering is a form of supervised learning.

Predictive clustering trees (PCTs) are tree structured models that generalize decision trees. The key properties of PCTs relevant to our approach are that (1) they can be used to predict many or all labels of an example at once (multi-target), (2) they can be applied to a wide range of prediction tasks (classification and regression) (3) they can handle examples represented by means of a complex representation (Džeroski et al. 2007), which is achieved by plugging in a suitable distance for the task at hand, and *iv)* their tree structure allows us to estimate and exploit the effect of autocorrelation in different ways at the different nodes of the tree (non-stationarity). In the context of this paper, PCTs are learned by plugging distances, which exploit the network structure, in the PCTs induction and obtaining predictive models that will be able to deal with autocorrelated data. This is done by maximizing the variance reduction and maximizing cluster homogeneity (in terms of autocorrelation) at the same time when evaluating the candidates for adding a new node to the tree, thus improving the predictive performance of the obtained models.

The network setting that we address in this work is based on the use of both the descriptive information (node attributes) and the network structure during training. We only use the descriptive information in the testing phase, where we disregard the network structure. More specifically, in the training phase, we assume that all examples are labeled and that the given network is complete. In the testing phase, all testing examples are unlabeled and the network is not given. The fact that the network is not necessary in the testing phase, can be very beneficial, especially in cases where predictions needs to be made for new examples for which connections to other examples are not known or need to be confirmed.

The setting where a network is given with some nodes labeled and some nodes unlabeled can be mapped to our setting. Namely, we can use the nodes with labels and the projection of the network on these nodes for training. We can then use only the unlabeled nodes without network information in the testing phase.

Our network setting is very much different from existing approaches to network classification and regression, where typically the descriptive information is in a tight connection to the network structure. The connections (edges in the network) between the data in the training/testing set are predefined for a particular instance, and are used to generate the descriptive information associated to the nodes of the network (see, for example, Steinhaeuser et al. (2011)). Therefore, in order to predict the value of the response variable(s), besides the descriptive information one needs the connections (edges in the network) to the related/similar entities.

Our network setting is also very much different from what is typically done in network analysis, where the general focus is on exploring the structure of a network by calculating its properties (e.g., the degrees of the nodes, the connectedness within the network, scalability, robustness, etc.). The network properties are then fitted into an already existing mathematical (theoretical) network (graph) model (Steinhaeuser et al. 2011).

From the descriptive perspective, the tree models obtained by the proposed algorithm allow us to obtain a hierarchical view of the network, where clusters can be employed to design a federation of hierarchically arranged networks. This can be useful, for instance, in wireless sensor networks, where a hierarchical structure is one of the possible ways to reduce the communication cost between the nodes (Li et al. 2007b). Moreover, it is possible to browse the generated clusters at different levels of the hierarchy, where each cluster can naturally consider different effects of the autocorrelation phenomenon on different portions of the network: at higher levels of the tree, clusters will be able to consider autocorrelation phenomenon that are spread all over the network, while at lower levels of the tree, clusters will consider local effects of autocorrelation. In this way, we can consider non-stationary autocorrelation.

From the predictive perspective, according to the splitting tests in the tree, it is possible to associate an observation (testing node of a network) to a cluster. The predictive model associated to the cluster can then be used to predict its response value (or response values, in the case of multi-target tasks).

The contributions of this paper are in

- the investigation of the different autocorrelation measures introduced in the literature and their use in the case of network structured data;
- the development of an approach that uses these autocorrelation measures in the induction of PCTs by taking into account variations in the global/ local data distribution across the network;
- a theoretical discussion based on specific properties of autocorrelation, that are exploited in PCTs induction, and allow the discovery of clusters of dense zones of the network with similar values in the response variable;
- an extensive evaluation of the effectiveness of the proposed approach on regression problems (single- or multi-target) in real network data and network data derived from spatial data.

The algorithm proposed in this paper extends the predictive clustering framework implemented in the CLUS system (Blockeel et al. 1998)[1]. It allows CLUS to handle network data in the setting outlined above. Given a fully described network (nodes and edges), we evaluate our algorithm on real-world data networks (among them several geographical data networks), comparing it to approaches that do not take into account the global and local dependencies into the network.

The paper is based on the preliminary work by Stojanova et al. (2011a). However, this paper significantly extends and upgrades the work presented there. We first proposed an approach (Stojanova et al. 2011b) that deals with the global and local effects of the spatial autocorrelation in PCTs. This was followed by our preliminary work on considering network autocorrelation in PCTs (Stojanova et al. 2011a). The work presented here significantly extends the latter in the following directions:

- Motivation for this work is given, both from the theoretical and application perspective.
- An extensive discussion of related work in Collective Inference, Transductive and Semi-Supervised Learning, as well as Predictive Clustering is given.
- A theoretical discussion is given, based on the specific properties of autocorrelation and on how these properties are exploited in the proposed extension of PCT induction.
- We consider the network regression task in a multi-target formulation.
- We present new experiments on additional datasets, including real data about social networks.
- Additional experiments on social and spatial data networks that empirically confirm the considerations reported in the theoretical discussion and show how our algorithm is able to capture autocorrelation within the learned PCTs.
- We also report new experiments with a multi-target network regression task that illustrate how our algorithm adequately combines the effect of autocorrelation for several response variables in the case of learning Multi-Target multi-target PCTs.

[1] The CLUS system is available at http://www.cs.kuleuven.be/~dtai/clus.

3 Related work

The motivation for this work comes from research reported in the literature on mining network data and predictive clustering. In the following subsections, we discuss background and related work from both research fields.

3.1 Mining network data

Numerous approaches have been designed for modeling a partially labeled network and providing accurate estimates of unknown labels associated with the unlabeled nodes. These approaches have been mainly studied in the research fields of collective inference, active inference, semi-supervised and transductive inference. Details on the principal studies in these fields are described below.

In collective inference, interrelated values are inferred simultaneously and estimates of neighboring labels influence one another (Macskassy and Provost 2007; Gallagher et al. 2008; Sen et al. 2008). Exact inference in this context is known to be an NP-hard problem (Cooper 1990) and there is no guarantee that network data satisfies the conditions that make exact inference tractable for collective inference. Thus, most of the research in collective inference has been devoted to the development of approximate learning algorithms.

Popular approximate inference algorithms include iterative inference, Gibbs sampling, loopy belief propagation and mean-field relaxation labeling. An outline of the strengths and weakness of these algorithms is reported in Sen et al. (2008). In general, one of the major advantages of collective inference lies in its powerful ability to learn various kinds of dependency structures (e.g., different degrees of correlation (Jensen et al. 2004)). However, as pointed out by Neville and Jensen (2007), when the labeled data are very sparse, the performance of collective classification might be largely degraded due to the insufficient number of neighbors. This is overcome by incorporating informative "ghost edges" into the network to deal with sparsity issues (Macskassy 2007; Neville and Jensen 2007). An alternative solution to the sparsity of labels is provided by Bilgic and Getoor (2008), where the authors resort to an active learning approach in order to judiciously select nodes for which a manual labeling is required from the domain expert.

Interestingly, learning problems similar to the predictive tasks arising in network learning have been recently addressed outside the areas of network learning and graph mining. In particular, in the area of in semi-supervised learning and transductive learning (Vapnik 1998), a corpus of data without links is given to the algorithms. The basic idea is to connect the data into a weighted network by adding edges (in various ways) based on the similarity between entities and to estimate a function on the graph which guarantees the consistency with the label information and the smoothness over the whole graph (Zhu et al. 2003). The constraint on smoothness implicitly assumes positive autocorrelation in the graph, that is, nearby nodes tend to share the same class labels (i.e., homophily).

Rahmani et al. (2010) use a transductive learning approach in order to classify the function of proteins in protein–protein interaction (PPI) networks by using only data

related to the structure of the networks. The classification task aims at predicting the particular function associated with a node (protein). Appice et al. (2009) address the task of the network regression with a transductive approach that follows the main idea of iterative inference described in Sen et al. (2008). Their regression inference procedure is based on a co-training approach according to which separate model trees are learned from the attributes of a node and the attributes aggregated in the neighborhood of the nodes. During an iterative learning process, each of these trees is used to label the unlabeled nodes for the other. The learning process is robust to sparse labeling and low label consistency and improves traditional model tree induction across a range of geographical data networks. As with other transductive learning approach, no final model is produced. This idea of using aggregated relational features to enhance the node-wise similarity measurement is popular in network mining and it is not restricted to the transductive setting. In fact, Popescul and Ungar (2003) and Hasan et al. (2006) have demonstrated that using aggregated relational features in an iterative way can be quite effective not only for the collective classification, but also for the link prediction task. The construction of aggregated features has been also investigated in Grcar and Lavrac (2011), where the authors present an efficient classification algorithm for categorizing video lectures in heterogeneous document information networks. They first transform documents into bag-of-words vectors, then decompose the corresponding heterogeneous network into separate graphs and compute structural-context feature vectors, and finally construct a common feature vector space to be used in the mining phase.

The studies described above investigate several means (from the synthesis of in-network features to the propagation of a response across the network) to account for autocorrelation and improve the accuracy of prediction. However, they only consider the cases where training and testing data (nodes) belong to the same network. This means that the prediction phase requires complete knowledge of the network arrangement (e.g., connections to other nodes of the network) of any unlabeled node to be predicted. In this paper, we are inspired by these works to consider the network structure in learning data with autocorrelation, but make a step further and attempt to learn a general predictive model, which allows the labeling of nodes also in the case connections to other nodes are not known or need to be confirmed.

In the context of social networks, Weng et al. (2010) proposed TwitterRank, an algorithm for finding topic-sensitive influential *twitterers* which is based on the assumptions that the presence of homophily implies that there are *twitterers* who are selective in choosing *friends* to follow. Next, Kwak et al. (2010) investigated homophily in two contexts: geographic location and popularity. They considered the time zone of a user as an approximate indicator for the location of the user and the number of followers as a measure for user's popularity. Among reciprocated users they observed some level of homophily.

We consider the work of Steinhaeuser et al. (2011), which follows an approach close to the idea of predictive clustering. In fact, Steinhaeuser et al. (2011) combine a descriptive data mining task (clustering) with a predictive data mining task (regression) and argue that using networks as a data representation provides a unified framework for identifying and characterizing patterns in climate data. In their case, the network is built a-posteriori on the basis of the values of the Pearson's correlation coefficients

between pair of nodes on time series collected for the same variable. Clustering then boils down to a community detection problem that allows the proposed approach to group interconnected nodes so that the pairwise walking distance between two nodes in the cluster is minimized. Finally, the prediction is obtained by means of a linear regression model built on the cluster to which the nodes to be predicted belong (spatially). The innovative aspects of our proposal with respect to this work are manifold. First, clustering is addressed as a hierarchical task. Second, the correlation is measured on nodes which are interconnected in an existing network, while Steinhaeuser et al. (2011) measure the correlation to define virtual edges between examples. Finally, in our idea, clusters are not constrained from the structure of the training network.

3.2 Mining predictive clustering trees

PCTs (Blockeel et al. 1998) view decision trees as a hierarchy of clusters: the top-node corresponds to one cluster containing all data, which is recursively partitioned into smaller clusters while moving down the tree. The task of mining predictive clustering trees can be formalized as follows:
Given

- a descriptive space $\mathbf{X} = \{X_1, X_2, \ldots X_m\}$ spanned by m independent (or predictor) variables X_j, which can be both continuous and discrete;
- a target space $\mathbf{Y} = \{Y_1, Y_2, \ldots, Y_q\}$ spanned by q dependent (or target) variables Y_j;
- a set T of training examples, (x_i, y_i) with $x_i \in \mathbf{X}$ and $y_i \in \mathbf{Y}$.

Find a tree structure τ which represents:

- A set of hierarchically organized clusters on T such that for each $u \in T$, a sequence of clusters $C_{i_0}, C_{i_1}, \ldots, C_{i_r}$ exist for which $u \in C_{i_r}$ and the containment relation $C_{i_0} \supseteq C_{i_1} \supseteq \ldots \supseteq C_{i_r}$ is satisfied. Clusters $C_{i_0}, C_{i_1}, \ldots, C_{i_r}$ are associated to the nodes $t_{i_0}, t_{i_1}, \ldots, t_{i_r}$, respectively, where each $t_{i_j} \in \tau$ is a direct child of $t_{i_{j-1}} \in \tau$ $(j = 1, \ldots, r)$ and t_{i_0} is the root of the structure τ.
- A predictive piecewise function $f : \mathbf{X} \rightarrow \mathbf{Y}$, defined according to the hierarchically organized clusters. In particular,

$$\forall u \in \mathbf{X},\ f(u) = \sum_{t_i \in leaves(\tau)} D(u, t_i) f_{t_i}(u) \tag{1}$$

where $D(u, t_i) = \begin{cases} 1 \text{ if } u \in C_i \\ 0 \text{ otherwise} \end{cases}$, $f_{t_i}(u)$ is a (multi-target) prediction function associated to the leaf t_i and $D(u, t_i)$ is only associated to leaves. Each $f_{t_i}(u)$ is a vector of q constant values.

Clusters are identified according to both the descriptive space and the target space $\mathbf{X} \times \mathbf{Y}$ (Fig. 1c). This is different from what is commonly done in predictive modelling (Fig. 1a) and classical clustering (Fig. 1b), where only one of the spaces is typically considered.

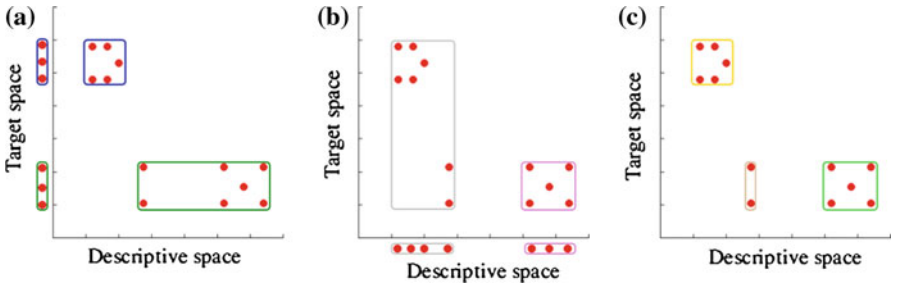

Fig. 1 Illustration of predictive clustering: **a** clustering in the target space, **b** clustering in the descriptive space, and **c** clustering in both the target and descriptive spaces. Note that the target and descriptive spaces are presented here as one-dimensional axes for easier interpretation, but can actually be of higher dimensionality

Note that this general formulation of the problem allows us to take into account two different aspects:

- the clustering phase can consider the (possibly) complex nature of the data such as, in our extension, the network structure;
- the prediction model mining phase can consider multiple target variables $\mathbf{Y} = Y_1, Y_2, \ldots, Yq$ at the same time.

The construction of a PCT is not very different from the construction of standard decision tree (see, for example, the C4.5 algorithm proposed by Quinlan (1993)): at each internal node t, a test has to be selected according to a given evaluation function. The main difference is that for PCTs, we select the best test by maximizing the (inter-cluster) variance reduction, defined as:

$$\Delta_Y(C, \mathcal{P}) = Var_Y(C) - \sum_{C_k \in \mathcal{P}} \frac{|C_k|}{|C|} Var_Y(C_k) \tag{2}$$

where C represents the cluster associated with t and $\mathcal{P}$ defines the partition $\{C_1, C_2\}$ of C. The partition is defined according to a Boolean test on a predictor variable in $\mathbf{X}$. By maximizing the variance reduction, the cluster homogeneity is maximized, improving at the same time the predictive performance. $Var_Y(C)$ is the variance of the target variable Y in the cluster C.

If the variance $Var(\cdot)$ and the predictive function $f(\cdot)$ are considered as parameters, instantiated for the specific learning task at hand, it is possible to easily adapt PCTs to different domains and different tasks. The PCT framework allows different definitions of appropriate variance (dispersion) functions for different types of data and can thus handle complex structured data as targets. To construct a regression tree, for example, the variance function returns the variance of the response values on the examples in a cluster, whereas the predictive function $f_{t_i}(u)$ is the average of the response values in a cluster. Indeed, by appropriately defining the variance and predictive functions, PCTs have been used for clustering (Blockeel et al. 1998), multi-target classification and regression (Blockeel et al. 1998; Demšar et al. 2005), and time series data analysis (Džeroski et al. 2007). Because PCTs can work with complex structured data by

introducing adequate variance measures, PCTs are a good candidate to appropriately deal with data that are not i.i.d., where the complexity comes from the need of taking autocorrelation into account.

In this paper, we consider an extended version of the problem of constructing PCTs for multi-target regression problems and, in particular, we consider the network structure in addition to the descriptive and target spaces.

4 Learning PCTs from network data

This section is devoted to the description of the proposed algorithm. In particular, we first formalize the problems to be solved and then we describe the considered autocorrelation measures and provide a detailed description of the implemented algorithm. In the last part, we give an analysis of the time complexity and discuss some theoretical issues raised by the specific properties of autocorrelation.

4.1 The problem

A network is a set of entities connected by edges. Each entity is called a node of the network. A number (which is usually taken to be positive) called "weight" is associated with each edge. In a general formulation, a network can be represented as a (weighted and directed) graph, i.e., a set of nodes and a ternary relation which represents both the edges between nodes and the weight associated to each edge. Formally, a data network G is a pair (V, E), where:

- V is a set of nodes, and
- E is a set of weighted edges between nodes, that is, $E \subseteq \{\langle u, v, w\rangle | u, v \in V, w \in \mathbb{R}^+\}$.

The network G is represented by an adjacency matrix $\mathbf{W}$ with entries $w_{ij} > 0$ if there is an edge connecting i to j, and $w_{ij} = 0$ otherwise. We impose $w_{ii} = 0$ and we define the degree of a node u_i as $Deg(u_i) = \sum_j w_{ij}$. Figure 2 shows an example of a data network, where different colors represent different node labels.

In practice, when the original data comes in the form of a network, the weights w_{ij} represent the strength of the connections from one node to another and usually have a natural interpretation. For example, in hypermedia data, it could be the number of hyperlinks from one page to another or, in Twitter, the fact that a user follows another user (keeping in mind that a user is not obligated to reciprocate followers by following them). These examples, indeed, are related to the case of modeling asymmetric relationships. In the case of modeling symmetric relationships, it could be the frequency with which a pair of lectures have been viewed together or a binary value indicating whether two proteins interact each other. When the weights are not readily available from the data, they are often computed based on symmetric and nonnegative similarity measures between nodes, based on some node properties, i.e., l_i and l_j.

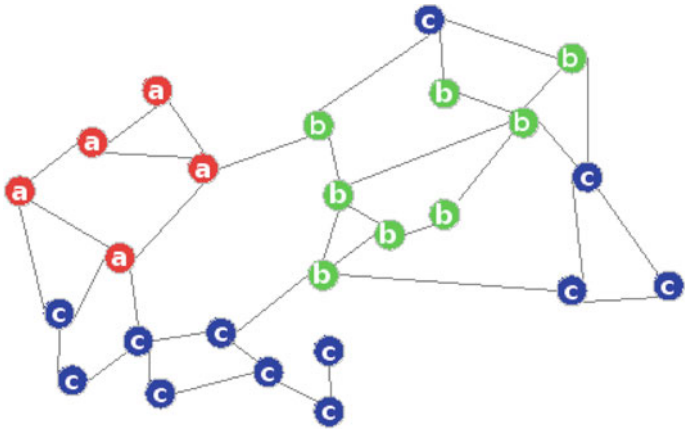

Fig. 2 Autocorrelation in network data. Different labels are given in *different colors*

A popular choice is to use the Gaussian-like similarity measure:

$$w_{ij} = \begin{cases} e^{-\frac{diss(l_i,l_j)^2}{b^2}} & \text{if } diss(l_i,l_j) < b \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $diss(l_i, l_j)$ represents the dissimilarity between node i and node j, while b is referred to as the bandwidth. In this way, nodes with dissimilarity greater than b are treated as disconnected nodes.[2]

If l_i and l_j are exactly the same, the weighting function w_{ij} for these nodes will be one. In other cases, the weight w_{ij} will decrease according to a Gaussian-like function as the dissimilarity between i and j increases. If nodes i and j have highly dissimilar values, the weight will be approximately zero. Using the bandwidth b, it is possible to consider only the nodes whose values are most similar.

As an alternative, it is possible to use a discrete weighting function (see Eq. (4)) and a bisquare density function (see Eq. (5)):

$$w_{ij} = \begin{cases} 1 - \frac{diss(l_i,l_j)}{b} & \text{if } diss(l_i,l_j) < b \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$w_{ij} = \begin{cases} \left(1 - \frac{diss(l_i,l_j)^2}{b^2}\right)^2 & \text{if } diss(l_i,l_j) < b \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

which we refer to as "Euclidean" and "Modified", respectively. Indeed, as empirically shown by Stojanova et al. (2011a), the three weighting functions show very similar results. For this reason, in the remainder of this paper, we will always use the Euclidean weighting function. If each node can be represented in Euclidean space R_d (as is for

[2] At this point, we do not assume any information on the type of the considered properties and on the dissimilarity functions.

instance the case with spatial data), $diss(l_i, l_j) = \| l_i - l_j \|$ where $l_i \in R_d$ ($l_j \in R_d$) describes the location of the node i (j). Whatever weighting schema is employed, the choice of the bandwidth b plays a crucial role. This means that the main problem is how to select the optimal bandwidth b. This problem is described and tackled in Sect. 4.3.1.

In this work, each node of the network is associated with a data observation $(x, y) \in \mathbf{X} \times \mathbf{Y}$ where $\mathbf{X} = \{X_1, X_2, \ldots, X_m\}$ (each X_i can be either continuous or discrete) and $\mathbf{Y} = \{Y_1, Y_2, \ldots, Y_q\}$ is the possibly unknown, possible multi-dimensional response variable with a range in $\mathbb{R}^q$. In order to formalize the learning task we are referring to, we need to define the network arrangement of the data with the goal of explicitly taking autocorrelation into account when learning the multi-target prediction function. For this purpose, in addition to the descriptive space $\mathbf{X}$ and the target space $\mathbf{Y}$, it is necessary to add information on the connections within the network (e.g., the links between the objects involved in the analysis and the pairwise dissimilarities between them) for the training data.

The network regression problem that we address can now be formulated as follows. Given:

1. a network $G = (V, E)$ where V is the set of nodes and $E \subseteq \{\langle u, v, w\rangle | u, v \in V, w \in \mathbb{R}^+\}$ is the set of edges;
2. a function $\eta : V \longmapsto (\mathbf{X} \times \mathbf{Y})$ which associates each node with its predictor and response values.

Find:

A PCT which represents a multi-dimensional (or multi-target) piecewise predictive function $f : \mathbf{X} \to \mathbf{Y}$ defined according to hierarchically organized extracted clusters, such that both the inter-cluster variance on the response variable is minimized and the inter-cluster autocorrelation on the response variable in maximized.

4.2 Network autocorrelation measures

When taking autocorrelation into account, we consider three alternative autocorrelation measures within the PCTs induction algorithm.

The first measure we consider is the Global Moran's I (Cressie 1993), which is borrowed from spatial data analysis, but also fits the general structure of network data. This measure requires a weight matrix that reflects the intensity of the relationships (the strength of the connection) between connected nodes. Formally, the **Global Moran's** I is defined as:

$$I_Y(C) = \frac{N_C}{\sum_{v_i \in C} \sum_{v_j \in C} w_{ij}} \frac{\sum_{v_i \in C} \sum_{v_j \in C} w_{ij}(Y_i - \overline{Y_C})(Y_j - \overline{Y_C})}{\sum_{v_i \in C} (Y_i - \overline{Y_C})^2} \tag{6}$$

where C is a cluster of nodes; N_C is the cardinality of C; Y_i and Y_j are the values of the variable Y for the nodes v_i and v_j according to the function η, respectively; Y is the target variable of interest; $\overline{Y_C}$ is the mean of Y in C; and $w_{i,j}$ are the values of

the adjacency matrix for the weight associated to (v_i, v_j). Dubin (1998) shows that the expected value of Moran's I (calculated assuming the values of Y are distributed randomly) is

$$E(I_Y) = \frac{-1}{N_C - 1} \tag{7}$$

where $I_Y(C) \geq -1/(N_C - 1)$ indicates positive autocorrelation, while $I_Y(C) \leq -1/(N_C - 1)$ indicates negative autocorrelation. The values of $I_Y(C)$ generally range from -1 to +1 and high positive (negative) values of $I_Y(C)$ indicate strong positive (negative) autocorrelation.

Relational Autocorrelation (RA) is an alternative measure of network autocorrelation used in collective classification to estimate the strength of statistical dependencies of the values of a variable Y between linked nodes (Angin and Neville 2008). Any traditional measure of association, such as the χ^2 statistic or information gain, can be used to assess the association between interconnected values of Y. One possibility is to use a variant of the Pearson's correlation coefficient that measures the correlation of a continuous variable Y with itself:

$$RA_Y(C) = \frac{\sum_{\langle v_i, v_j, w_{ij} \rangle \in E} (Y_i - \overline{Y_C})(Y_j - \overline{Y_C})}{\sum_{\langle v_i, v_j, w_{ij} \rangle \in E} (Y_i - \overline{Y_C})^2}. \tag{8}$$

As in Global Moran's I, high positive values indicate positive autocorrelation, while high negative values indicate negative autocorrelation. Both Global Moran's I and Relational Autocorrelation allow us to measure autocorrelation of a variable Y across the network.

Global Moran's I takes into account the weight associated to each edge, while Relational Autocorrelation only considers the existence or absence of an edge connecting two nodes and neglects the strength associated to the edge. Therefore, the RA can be considered a special case of Global Moran's I where the weights are binary and each edge is labeled with a weight of one.

A different measure is the **Connectivity (Randic) Index** (CI) which is limited to the consideration of the structure of a network (or graph). In this paper we exploit the extension of the classical Randic connectivity index (Randic 1998) for the case of weighted networks (Ghimire et al. 2008). Formally, the Connectivity Index is defined as:

$$CI(C) = \sum_{v_i \in C} \sum_{v_j \in C} \frac{w_{ij} + w_{ji}}{2\sqrt{Deg(v_i)Deg(v_j)}} \tag{9}$$

where $Deg(v_i)$ and $Deg(v_j)$ represent the degrees of nodes v_i and v_j respectively. CI is an indication of the connectedness (or branching) of a cluster C and can be used to compare the connectivity among clusters. It is typically used in chemistry, since it can be well correlated with a variety of physico-chemical properties of alkanes, such as boiling points, surface area and solubility in water.

High values of $CI(C)$ indicate high connectivity, while low values of $CI(C)$ indicate low connectivity (Ghimire et al. 2008). As we assume a network where all connected nodes are directly linked, measuring connectivity is also a way to quantify correlation in the network.

Finally, we point out that in all presented measures, only the possible influence of a node linked directly to the considered node is taken into account. In other words, these autocorrelation measures consider only pairs of directly connected nodes. In future work, it is possible to define novel autocorrelation measures that, differently from the measures available in the literature, also take indirect connections into account when computing autocorrelation.

4.3 The algorithm

We can now proceed to describe the top-down induction algorithm for building PCTs from network data (Algorithm 1). It is a recursive algorithm which takes as input the network $G = (V, E)$ and the function $\eta: V \mapsto \mathbf{X} \times \mathbf{Y}$ and partitions the set of nodes V until a stopping criterion is satisfied (Algorithm 1 line 2). Since the implementation of this algorithm is based on the implementation of the CLUS algorithm, we will call this algorithm NCLUS (for Network CLUS).

Algorithm 1 Top-down induction of NetworkPCTs

1: **procedure** NCLUS($G = (V, E), \eta(\cdot)$) **returns** tree
2: **if** stop($V, \eta(\cdot)$) **then**
3: **return** leaf(Prototype($V, \eta(\cdot)$))
4: **else**
5: $(c^*, h^*, \mathcal{P}^*, \mathcal{P}_\mathcal{V}{}^*) = (null, 0, \emptyset, \emptyset)$
6: $C = \{\eta(v) | v \in V\}$
7: **for each** possible Boolean test c according to values of $\mathbf{X}$ on C **do**
8: $\mathcal{P} = \{C_1, C_2\}$ partition induced by c on C
9: $\mathcal{P}_\mathcal{V} = \{V_1, V_2\}$ = partition induced by $\mathcal{P}$ on V;
10: $h = \frac{\alpha}{q} \sum_{Y \in \mathbf{Y}} \Delta_Y(C, \mathcal{P}) + \frac{(1-\alpha)}{q} \sum_{Y \in \mathbf{Y}} A_Y(G, \eta(\cdot), \mathcal{P})$
11: **if** $(h > h^*)$ **then**
12: $(c^*, h^*, \mathcal{P}^*, \mathcal{P}_\mathcal{V}{}^*) = (c, h, \mathcal{P}, \mathcal{P}_\mathcal{V})$
13: **end if**
14: **end for**
15: $\{V_1, V_2\} = \mathcal{P}_\mathcal{V}{}^*$
16: $tree_1$ = NCLUS($(V_1, E), \eta(\cdot)$)
17: $tree_2$ = NCLUS($(V_2, E), \eta(\cdot)$)
18: **return** node($c^*, tree_1, tree_2$)
19: **end if**

The main loop (Algorithm 1, lines 7–14) searches for the best attribute-value test c^* that can be associated to a node t. The algorithm associates the best test c^* to the internal node t and calls itself recursively to construct a subtree for each subnetwork in the partition $\mathcal{P}_\mathcal{V}{}^*$ induced by c^* on the training nodes.

Possible tests are of the form $X \leq \beta$ for continuous attributes, and $X \in \{x_{i_1}, x_{i_2}, \ldots, x_{i_o}\}$ (where $\{x_{i_1}, x_{i_2}, \ldots, x_{i_o}\}$ is a subset of the domain Dom_X of X) for discrete attri-

butes. For continuous attributes, possible values of β are found by sorting the distinct values of X in the training set associated to t, then considering a threshold between each pair of adjacent values. Therefore, if the cases in t have d distinct values for X, at most $d-1$ thresholds are considered.

For discrete attributes, possible subsets of values are selected by relying on a non-optimal greedy strategy (Mehta et al. 1996). Starting with an empty set $Left_t = \oslash$ and a full set $Right_t = Dom_X$, where Dom_X is the domain of X, this iterative procedure moves one element from $Right_t$ to $Left_t$, such that the move results in an increased reduction of variance for the target variable Y. This differs from the classical solution by Breiman et al. (1984), where some ordering on the possible values of Dom_X is defined apriori, according to the data distribution. However, the classical solution cannot deal with multi-target prediction tasks as PCTs can. If the examples in t have d distinct (discrete) values, $\sum_{i=2}^{d-1} i = \frac{d^2-3d}{2}$ splits are considered.

The algorithm evaluates the best split according to the formula (10) reported in Algorithm 1, line 10.

$$h = \frac{\alpha}{q} \sum_{Y \in \mathbf{Y}} \Delta_Y(C, \mathcal{P}) + \frac{(1-\alpha)}{q} \sum_{Y \in \mathbf{Y}} A_Y(G, \eta(\cdot), \mathcal{P}) \tag{10}$$

This formula is a linear combination of the variance reduction $\Delta_Y(C, \mathcal{P})$ and the autocorrelation measure $A_Y(G, \eta(\cdot), \mathcal{P})$.

Both, variance and autocorrelation are computed for the Y variable over the cluster C. In the case of multiple target variables, the average values of both, variance reduction $\Delta_Y(C, \mathcal{P})$ and autocorrelation $A_Y(G, \eta(\cdot), \mathcal{P})$ are taken over the set of target variables, where each target variable contributes equally to the overall h value.

The influence of these two parts of the linear combination when building the PCTs is determinated by a user-defined coefficient α that falls in the interval [0, 1]. When $\alpha = 0$, NCLUS uses only autocorrelation, when $\alpha = 0.5$ it weights equally variance reduction and autocorrelation, and when $\alpha = 1$ it ignores autocorrelation and works such as the original CLUS algorithm.

According to the above discussion on network autocorrelation measures, $A_Y(G, \eta(\cdot), \mathcal{P})$ can be defined in terms of each of the three indexes we introduced (Moran's I, RA and CI). However, since they all range in different intervals (but are consistently monotonic), it is necessary to appropriately scale them. Since variance reduction is non-negative, we decided to scale them both to the interval [0, 1], where 1 means high positive autocorrelation and 0 means high negative autocorrelation. The choice of the scaling interval does not affect the heuristic computation, therefore other scaling intervals are possible as well, provided that, in all cases, the same scaling is performed and the monotonicity of the scaled measure is maintained.

For example, for Moran's I,$A_Y(G, \eta(\cdot), \mathcal{P})$ is defined as:

$$A_Y(G, \eta(\cdot), \mathcal{P}) = \sum_{C_k \in \mathcal{P}} \frac{|C_k|}{|C|} \widehat{I}_Y(C_k) \tag{11}$$

where $\widehat{I_Y}(D_k)$ is the scaled Moran's I computed on D_k.

Moreover, in order to guarantee a fair combination of the variance reduction and the autocorrelation statistic $A_Y(G, \eta(\cdot), \mathcal{P})$, we also need to scale the variance reduction to the interval [0, 1]. For that purpose, we use a common scaling function:

$$\widehat{\Delta_Y}(C, \mathcal{P}) = \frac{\Delta_Y(C, \mathcal{P}) - \Delta min}{\Delta max - \Delta min} \tag{12}$$

where Δmax and Δmin are the maximum and the minimum values of $\Delta_Y(C, \mathcal{P})$ for a particular split.

The search stops when one of the two defined stopping criteria is satisfied. The first criterion stops the search when the number of examples in a leaf is smaller than $\sqrt{N}$, which is considered a good locality threshold that does not lose too much in accuracy (also for rule based classifiers) (Gora and Wojna 2002). The second criterion uses the statistical F-test to check whether a given split/test in an internal node of the tree results in a reduction in $\Delta_Y(C, \mathcal{P})$ that is statistically significant at a given significance level. Actually, the F-test is only theoretically correct for normally distributed populations. Since this assumption may not hold in this case, the usage of the F-test should here be considered, coherently with (Blockeel et al. 1998), as a heuristic for deciding when to stop growing a branch, not as a real statistical test. To choose the optimal significance level among the values in the set {1, 0.125, 0.1, 0.05, 0.01, 0.005, 0.001}, we optimize the MSE obtained with an internal 3-fold cross validation. When the first stopping criterion is not satisfied, we evaluate the second criterion. In the case one of the two stopping criteria is satisfied, the algorithm creates a leaf and labels it with a predictive function $f_{t_i}(u)$ (in this case the average of the response variable(s)) defined over the examples falling in that leaf. When predicting multiple response variables, the predictive function returns the vector of the averages of the responses values.

In NCLUS, a pruning strategy to prevent trees from over-fitting data is implemented. This strategy is the pessimistic error pruning strategy, which is also implemented in several regression/ model tree learners (including M5' and CLUS). According to this strategy, a subtree is kept only if the error at the leaf is greater than the error of the subtree multiplied by a scaling factor, which takes into account the statistical support and the complexity of the model (Wang and Witten 1997). The results that we present in this paper are those of the pruned tree models learned by NCLUS.

4.3.1 Choosing the bandwidth

The choice of the bandwidth (denoted by b in (3)) is perhaps the most critical decision to be taken in the modeling process. This parameter controls the degree of smoothing, with larger bandwidths causing stronger smoothing. An oversmoothed model will predict similar values of the target variable all across the network, while an undersmoothed model will predict values with so much local variation that it would be difficult to determine whether there are any patterns at all. At the same time, this parameter influences the calculation of autocorrelation.

The bandwidth may be defined manually or by using some adaptive method on the whole training network. For example, GWR (Fotheringham et al. 2002) explores

different values of the bandwidth and finds the one which yields the lowest cross-validated AIC (Akaike Information Criterion) value. In this study, for the selection of the bandwidth, we minimize the leave-one-out cross validated-(RMSE) Root Mean Square Error. Moreover, in this automatic determination of the bandwidth, the selection is not performed directly on the bandwidth b, but on $b^{\%}$, that is, the bandwidth expressed as a percentage of the maximum dissimilarity between two connected nodes. This means that the algorithm implicitly considers different bandwidth values b at different nodes of the tree depending on the maximum dissimilarity between connected examples falling in that node of the tree. The bandwidth $b^{\%}$ ranges in the interval [0, 100%].

Minimization is performed by means of the Golden section search (Brent 1973) that recursively partitions the $b\%$ domain. Golden section search is similar to binary search, improving it by splitting the range in two intervals with a length ratio of γ instead of 1 (equal parts). *Golden ratio* has the value $\gamma = \frac{1+\sqrt{5}}{2}$.

The share maintains a pair of minimum and maximum bandwidth values, $b_1^{\%}$ and $b_2^{\%}$ (at the first iteration, they are initialized as the minimum and maximum bandwidth in the interval [0, 100 %]). At each step, the algorithm identifies a point $b_3^{\%}$ between them, according to the golden ratio and computes the cross-validated error for that point ($error_{b_3^{\%}}$). ($b_1^{\%} < b_3^{\%} < b_2^{\%}$). The values of the function at these points are $f(b_1^{\%})$, $f(b_3^{\%})$, and $f(b_2^{\%})$ and, collectively, these are known as a "triplet". The algorithm than identifies the only parabola with a vertical axis that intersects the points $\{(b_1^{\%}, error_{b_1^{\%}}), (b_3^{\%}, error_{b_3^{\%}}), (b_2^{\%}, error_{b_2^{\%}})\}$. On the basis of the position of the minimum of this parabola, the system decides whether to consider ($b_1^{\%}$, $b_3^{\%}$) or ($b_3^{\%}$, $b_2^{\%}$) as the next pair of (minimum and maximum) $b^{\%}$ values.

The search stops when there is no reduction of cross-validated RMSE. In the algorithm, the RMSE is computed by fitting a weighted linear model for the example left out. Having decided to consider only the Euclidean weighting function (4), we optimize $b^{\%}$ only for this case.

4.4 Time complexity

The computational complexity of the algorithm depends on the computational complexity of adding a splitting node t to the tree, which in fact depends on the complexity of selecting a splitting test for t. A splitting test can be either continuous or discrete. In the former case, a threshold β has to be selected for a continuous variable. Let N be the number of examples in the training set; then the number of distinct thresholds can be $N-1$ at worst. They can be determined after sorting the set of distinct values. If m is the number of descriptive variables, the determination of all possible thresholds has a complexity $O(m * N * logN)$, assuming an optimal algorithm is used for sorting.

For each variable, the system has to compute the evaluation measure h for all the possible thresholds. This computation has, in principle, time-complexity $O((N-1) * (N + N * k))$, where $N-1$ is the number of thresholds, k is the average number of edges for each node in the network, $O(N)$ is the complexity of the computation of the variance reduction $\widehat{\Delta_Y}(C, \mathcal{P})$ and $O(N * k)$ is the complexity of the computation of autocorrelation $A_Y(G, \eta(\cdot), \mathcal{P})$. However, it is not necessary to recompute

autocorrelation values from scratch for each threshold, since partial sums in both variance reduction computation and in autocorrelation computation can be used. In particular, partial sums can be incrementally updated depending on the examples that are iteratively moved from the right to the left branch. This optimization makes the complexity of the evaluation of the splits for each variable $O(N * k)$. This means that the worst case complexity of creating a splitting node on a continuous attribute is $O(m * (NlogN + N * k))$.

Similarly, for a discrete splitting test (for each variable), the worst case complexity is $O((d - 1) * (N + N * k))$, where d is the maximum number of distinct values of a discrete variable ($d \leq N$). This complexity takes the same optimization as proposed for continuous splits into account.

Therefore, finding the best splitting node (either continuous or discrete) has a complexity of $O(m * (NlogN + N * k))+O(m * d * (N + N * k))$, that is $O(m * N * (logN + d * k))$, where m is the number of descriptive variables, N is the number of examples, d is the maximum number of distinct values of a discrete variable and k is the average number of edges for each example (for each node in the network).

4.5 Exploiting the properties of autocorrelation in NCLUS

The consideration of autocorrelation in clustering has been the subject of some recent work in spatial clustering (Glotsos et al. 2004) and network clustering (Jahani and Bagherpour 2011). Motivated by the demonstrated benefits of autocorrelation, we exploit some properties of autocorrelation to improve the quality of the PCTs.

The use of autocorrelation in predictive clustering offers several advantages since it allows to:

- determine the strength of the network effects on the variables in the model;
- consider tests on assumptions of stationarity and heterogeneity in the network;
- identify the possible role of the network interaction/ distance decay on the predictions associated to each of the nodes of the tree;
- focus on the "node neighborhood" to better understand the effects that it can have on other neighborhoods and vice versa.

These properties, identified by Arthur (2008), hold for spatial clustering. However, they also hold for the case of PCTs. Moreover, as recognized by Griffith (2003), autocorrelation implicitly defines a zoning of a (spatial) phenomenon and reduces the effect of autocorrelation in the prediction errors.

Variance reduction leads to more accurate models since it reduces the error on the training set. However, it does not have all the properties implicitly introduced by autocorrelation. This is due to the fact that variance reduction does not take the distribution of connections among the nodes in the network into account.

With regard to the statistical properties of the measures of autocorrelation, most of the theoretical research in Statistics and Econometrics exploits the so called "autoregressive model" in order to measure autocorrelation in networks (represented as a special case of a directed acyclic graph) (Griffith 2003). More formally, the autoregressive model is defined as:

$$\hat{e_i} = \rho \sum_j w_{ij}\, e_j + \epsilon_i \tag{13}$$

where $e_j = Y_j - \overline{Y}$ is the prediction error (where prediction is based on the average), ρ is a parameter that expresses the network dependence, w_{ij} are the elements of the neighborhood matrix W and the error ϵ_i follows a Gaussian (normal) distribution (Engle 1982).

In this case, the informal notion of network dependence is often implicitly based on an autoregressive framework, where the goal is to assess the predictive ability of the neighboring values of the data. As recognized by Li et al. (2007a), in order to informally assess the strength of the network dependence, exploratory data analysis should be based on estimating ρ in the autoregressive model (see Eq. 13). This means that the parameter ρ plays a crucial role in representing autocorrelation in the data.

One common solution for estimating ρ is to use a modified least squares estimator, which is the solution to the following quadratic equation in ρ:

$$\mathbf{e}^T(\mathbf{I} - \rho\mathbf{W})^T\mathbf{W}(\mathbf{I} - \rho\mathbf{W})\mathbf{e} = 0 \tag{14}$$

where $\mathbf{W}$ is the matrix representation of w_{ij}, $\mathbf{I}$ is the identity matrix and $\mathbf{e}$ is the vector of e_i values.

Although this estimator is consistent (Li et al. 2007a), its computation is not straightforward[3]. Therefore, instead of computing an estimate of ρ, Moran's I is commonly used in spatial data mining applications. Indeed, as proved by Jin (2010), under the assumption that $w_{ij} = w_{ji}$ ($\mathbf{W}$ is symmetric), Moran's I is monotonic in ρ. Unfortunately, this result is not valid when $w_{ij} \neq w_{ji}$ (some counterexamples can be found). Moreover, Li et al. (2007a) empirically proved that Moran's I is a good (not unbiased) estimator of ρ in the case when ρ approaches zero. This means that Moran's I is a good indicator of the network dependence under some conditions. The same conclusions can be drawn for the Relational Autocorrelation (RA) that is similar to the Moran's I, but considers the weights in a binary form. On the contrary, the Connectivity Index (CI) cannot be considered as a good estimator of ρ and, for this reason, we expect different results by varying the autocorrelation measure used in the predictive model. This analysis also confirms that a model that is able to take autocorrelation into account should lead to small values of ρ, that is, according to (13), it should lead to a reduction of the effect of an error in the neighborhood.

5 Empirical evaluation

Before we proceed to presenting empirical results, we provide a description of the used datasets and experimental settings[4].

[3] It would require the computation of the Maximum Likelihood Estimator of ρ which is impractical when large datasets need to be processed (Li et al. 2007a).

[4] All materials (datasets and system) are available at the following web page: http://kt.ijs.si/daniela_stojanova/DMKDpaper/

5.1 Datasets

In this experimental evaluation, we use four real network datasets obtained from social domains and six network datasets obtained from spatial data. They are described below.

5.1.1 Social network data

The **VideoL** dataset contains the ECML PKDD 2011 Discovery Challenge data (Antulov-Fantulin et al. 2011). The data are related to the content of VideoLectures.net, a free and open access multimedia repository of video lectures, mainly of research and educational character. The response is the total number of views of lectures published online, where pairs of lectures are viewed together (not necessarily consecutively) with at least two distinct cookie-identified browsers. The predictor variables include several properties of a lecture such as the type, category, author and language of the lecture, as well as the recorded and published dates of the lecture. Here we use the complete (training) data from the Challenge for 2009. The network structure has 754 nodes and 14398 edges. The nodes contain the lectures along with their properties, whereas the dissimilarities are the inverse of frequency (the number of distinct cookie-identified browsers) with which the respective pair of lectures was viewed together.

The **Books** dataset contains cross-rating book data from different users (Ziegler et al. 2005). For each node (book), the ISBN code, author, year of publication and publisher information are given, as well as the users' rating. The response is the average rating of all users. The network structure has 500 nodes (books) and 1167 edges. The nodes represent the books (described with their properties), whereas the weighted edges represent the dissimilarity (scalar distance) of the ratings given by the users to the respective pair of books.

The **Movies** datasets contains movie ratings given to movies by users of the online movie recommender service Movielens, collected during the period 1997–1998 [5]. Specifically, for each movie, it contains the IMDB movie identifier, genre, country, movie director and filming location, as well as all/top/audience critics's ratings: average scores, numbers of reviews/fresh scores/rotten scores from the Rotten Tomatoes film review aggregator. The response variable is the *all critics ratings*: all other ratings data are not included in the analysis. We are interested in pairs of movies that are ranked together by a single user, where the selected users had rated at least 20 movies. The network structure has 500 nodes and 202440 edges for the Movies1 dataset (*snapshot1*) and 500 nodes and 122748 edges for the Movies2 dataset (*snapshot2*). The nodes represent the movies (labeled with their properties), whereas the weighted edges represent the dissimilarity (scalar distance) of the ratings given by the users to the respective pair of movies.

The **Twitter** Maternal Health dataset contains the top Twitter users who recently mentioned maternal mortality. This dataset is obtained by a query performed on August 25, 2010 and it is sorted by betweenness centrality [6]. Specifically, it contains the

[5] http://www.grouplens.org/node/12.

[6] http://casci.umd.edu/NodeXL_Teaching/.

number of posts (tweets) of a user (response variable), user's registration date on Twitter and its time zone, as well as the number of tweets (posted by other users) that the user marked as "favorites" and the number of "following" and "followed" on Twitter. The relationships "following" and "followed" simply reflect the number of users that are subscribed to receive news information from a specific user and the number of users that a specific user is subscribed to receive news information from. Here, we address only the pairs of users that are in the relationship "following". The network structure has 109 nodes and 198 edges. The nodes are the users (along with their properties), whereas the weighted edges are the "following" relation between the Twitter users. Note that this relationship is binary ("1" if there is a relation "following" between two Twitter users and "0" otherwise) and we consider only edges contained in this relation.

The Twitter network differs from other communities networks (e.g., Facebook, MySpace and MSN), because it is an information and a community network, where a user connects to the latest news information about what he/she finds interesting. Note that Twitter relationships are not symmetric (i.e., networks are of directed nature), thus a path from a user to another user may follow different hops for which the inverse direction does not exist. For example, while in MSN a link represents a mutual agreement of a relationship, on Twitter a user is not obligated to reciprocate followers by following them.

5.1.2 Spatial network data

In the spatial datasets, the nodes are the spatial units of analysis considered (wards, sampling points, counties and forest fire sites). They are described by some attributes that will be discussed in detail below. The spatial units of analysis are at some distance apart in space.

The **NWE** (North-West England) dataset contains census data collected in the European project SPIN!. The data concerns North West England, an area that is decomposed into censual sections (wards). Census data provided by the 1998 Census are available at ward level. We consider the percentage of mortality (response variable) and measures of deprivation level in the ward according to index scores such as the Jarman Underprivileged Area Score, Townsend score, Carstairs score and the Department of the Environment Index, as well as the coordinates of the ward centroids. The nodes in the network structure are the 970 wards.

The datasets **SIGMEA_MS** and **SIGMEA_MF** (MS and MF) (Demšar et al. 2005) are derived from one multi-target dataset containing measurements of pollen dispersal (crossover) rates from two lines of plants (response variables), that is, the transgenic male-fertile (MF) and the non-transgenic male-sterile (MS) line of oilseed rape. The coordinates of each sample point are collected. The predictor variables are the cardinal direction and distance of the sample point from the center of the donor field, the visual angle between the sample plot and the donor field, and the shortest distance between the plot and the nearest edge of the donor field. The nodes in the network structures of both SIGMEA_MS and SIGMEA_MF are the 817 sampling points.

The **FOIXA** dataset (Debeljak et al. 2012) contains measurements of the rates of outcrossing at sample points located within a conventional field that comes from the surrounding genetically modified (GM) fields within a 400 ha large maize production area in the Foixa region in Spain. The measurements include the coordinates of the sampling points (units) and several predictor variables, that is, the number and size of the surrounding GM fields, the ratio of the size of the surrounding GM fields and the size of conventional fields and the average distance between the conventional and the GM fields. The nodes in the network structures of FOIXA represent 420 sampling points.

The **GASD** (USA Geographical Analysis Spatial Dataset) (Pace and Barry 1997) contains 3,106 observations on USA votes cast in 1980 presidential election per county. Besides the number of votes (response variable), the coordinates of the centroid of each county as well as the number of owner-occupied housing units, the aggregate income and the population over 18 years of age are reported, for the respective county. The 3106 counties represent the nodes in the network structure.

The **Forest Fires** (FF) dataset (Cortez and Morais 2007) is publicly available for research purposes from the UCI Machine Learning Repository [7]. It collects 517 forest fire observations from the Montesinho park in Portugal. The data, collected from January 2000 to December 2003, includes the coordinates of the forest fire sites, the burned area of the forest given in ha (response variable), the Fine Fuel Moisture Code (FFMC), the Duff Moisture Code (DMC), the Drought Code (DC), the Initial Spread Index (ISI), the temperature in degrees Celsius, the relative humidity, the wind speed in km/h and the outside rain in mm within the Montesinho park map. The nodes in the network structure represent the individual forest fires sites.

In the networks obtained from these spatial data, edges are defined for each pair of nodes and dissimilarities are computed according to the Euclidean distance between the nodes' spatial coordinates.

5.2 Experimental setup

5.2.1 Evaluation metrics

We evaluate the performance of several variants of NCLUS and compare it to the performance of the original CLUS algorithms, as well as to the performance of several other algorithms. The evaluation is performed on the two collections of datasets described above. The evaluation is performed in terms of several metrics, which include accuracy, model complexity, and learning time. In addition, we also measure autocorrelation of the errors of the learned models on the testing set, evaluating the ability of the different algorithms to capture the effect of autocorrelation within the learned models, as well as the correlation of the model predictions with the true target values on the test set.

All of these performance measures are estimated by using 10-fold cross validation. In particular, the accuracy is measured in terms of the RMSE, while the model

[7] http://archive.ics.uci.edu/ml/

complexity is measured in terms of the number of leaves in the learned trees. The computation time is measured in seconds. The experiments were run on an Intel Xeon CPU @2.00 GHz server running the Linux Operating System.

5.2.2 Algorithms compared

NCLUS is run with the automatic bandwidth determination, with the three different autocorrelation measures, that is, Moran's I (NCLUS_I), Relational Autocorrelation (NCLUS_RA) and Connectivity Index (NCLUS_CI), and with the user-defined parameter $\alpha = 0$ and $\alpha = 0.5$. NCLUS, with the above experimental configurations, is compared to the original CLUS algorithm (Blockeel et al. 1998).

Only for the spatial networks, where the spatial coordinates of each node are available in the data, NCLUS is also compared to a version of CLUS algorithm, where the spatial coordinates are treated as additional response variables. This is done only for the computation of the evaluation measures. In this way, we are able to implicitly take spatial autocorrelation into account. We refer to this configuration of CLUS as CLUS*.

Moreover, the empirical evaluation includes the well known tree-based method M5' Regression Trees (Quinlan 1993), as well as non tree-based methods such as Support Vector Regression (SVR) (Basak et al. 2007) and the k-Nearest Neighbors (k-NN) (Aha and Kibler 1991), which do not consider autocorrelation. The WEKA (Witten and Frank 2005) implementation of these algorithms was used with their default settings, with the number of examples in a leaf in M5', as well as the number of neighbors in k-NN, set to $\sqrt{N}$. Furthermore, we also compare NCLUS to the Iterative Transductive Learning (ITL) algorithm (Appice et al. 2009) that addresses the network regression problem for spatial data. ITL works in the transductive learning setting and considers autocorrelation. Finally, as a baseline (Base) we also use the default method that always predicts the mean (over the training set) of the response variable.

5.2.3 Statistical comparison

In order to compare the predictive capabilities of the learned models, we use the non-parametric Wilcoxon two-sample paired signed rank test (Orkin and Drogin 1990). To perform the test, we assume that the experimental results of the two methods compared are independent pairs $\{(q_1, r_1), (q_2, r_2), \ldots, (q_n, r_n)\}$ of sample data. We then rank the absolute value of the differences $q_i - r_i$. The Wilcoxon test statistics WT^+ and WT^- are the sum of the ranks from the positive and negative differences, respectively. We test the null hypothesis H_0: "no difference in distributions" against the two-sided alternative H_1: "there is a difference in distributions". Intuitively, when $WT^+ \gg WT^-$ and vice versa, H_0 is rejected. Whether WT^+ should be considered "much greater than" WT^- depends on the considered significance level. The null hypothesis of the statistical test is that the two populations have the same continuous distribution. Since, in our experiments, q_i and r_i are average MRSEs, $WT^+ \gg WT^-$ implies that the second method (R) is better than the first (Q). In all experiments reported in this empirical study, the significance level used in the test is set at 0.05.

Table 1 Comparison of error made by different learning approaches on social network data

Method/ Network dataset		VideoL	MOVIES1	MOVIES2	BOOKS	TWITTER
NCLUS_I	$\alpha = 0.0$	686.32	1.08	1.17	**2.53**	**10170.92**
NCLUS_I	$\alpha = 0.5$	653.2	**1.06**	**1.02**	**2.53**	**10170.92**
NCLUS_RA	$\alpha = 0.0$	686.32	1.08	1.17	**2.53**	**10170.92**
NCLUS_RA	$\alpha = 0.5$	653.2	**1.06**	**1.02**	**2.53**	**10170.92**
NCLUS_CI	$\alpha = 0.0$	686.32	1.62	2.11	**2.53**	**10170.92**
NCLUS_CI	$\alpha = 0.5$	653.2	1.51	1.31	**2.53**	10712.23
CLUS		660.69	1.53	2.42	2.83	10641.03
SVR		721.43	1.77	2.52	2.65	13875.73
k-NN		937.68	1.70	2.58	2.65	11007.23
M5'		**574.17**	2.09	2.2	2.67	12253.34
Base		722.39	2.10	2.70	**2.53**	13255.27

The RMSEs (estimated by 10-fold CV) of the models obtained with NCLUS_I, NCLUS_RA, NCLUS_CI, CLUS, SVR, *k*-NN and M5', as well as the default Base model. For each network dataset, the best results are highlighted in bold

5.3 Results and discussion

5.3.1 Social network data

Table 1 reports the average errors of the PCTs by NCLUS_I, NCLUS_RA, NCLUS_CI and CLUS, as well as the errors of the SVR, *k*-NN and M5' approaches, on the social network data. The last row in Table 1 gives the errors of the Base model that always predicts the mean. For each network dataset, the best results are highlighted in bold.

We can observe that the NCLUS error depends on the network autocorrelation measure and on the relative importance given to the variance reduction and autocorrelation when building the PCT (according to the α parameter). The use of different autocorrelation measures does not change the obtained results much. However, the best results, in line with considerations reported in Sect. 4.5, are obtained by using the Global Moran's I and the Relational Autocorrelation. In contrast, the α parameter significantly affects the errors. The best results are always obtained with $\alpha = 0.5$ (when $\alpha = 0$ NCLUS uses only autocorrelation, when $\alpha = 0.5$ NCLUS equally weights variance reduction and autocorrelation, and when $\alpha = 1$ NCLUS works such as the original CLUS algorithm).

The results of NCLUS_I and NCLUS_RA are very similar and reflect the covariance between pairs of observations. The only difference between the two measures is in the existence of weights in the definition of the Global Moran's I. The weights reflect the strength of the relations between the nodes in the network and are associated to the edges in the network. Thus, the difference in the two measures of autocorrelation is in the explicit (Euclidean similarity measure vs. binary) strength of the relation between the nodes in the network, which usually comes with the definition of the network. On the other hand, NCLUS_CI performs worse than NCLUS_I and NCLUS_RA. From the results presented in Table 1, we can see that NCLUS compares very well to

Table 2 Statistical comparison of the performance of different NCLUS variants on social network data

Network/Method Dataset	NCLUS_I		NCLUS_RA		NCLUS_CI	
	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 0$	$\alpha = 0.5$
VideoL	(−)0.33	(+)0.28	(−)0.33	(+)0.28	(−)0.33	(+)0.28
MOVIES1	(+)**0.01**	(+)**0.01**	(+)**0.01**	(+)**0.01**	(−)0.33	(+)0.58
MOVIES2	(+)**0.01**	(+)**0.01**	(+)**0.01**	(+)**0.01**	(+)**0.01**	(+)**0.01**
BOOKS	(+)**0.01**	(+)**0.01**	(+)**0.01**	(+)**0.01**	(+)**0.01**	(+)**0.01**
TWITTER	(+)0.96	(+)0.96	(+)0.96	(+)0.96	(+)0.96	(−)0.58

The p-values of the Wilcoxon signed rank test comparing NCLUS and CLUS. (+) means that NCLUS is better than CLUS (i.e. $WT^+ > WT^-$), (−) means that CLUS is better than NCLUS (i.e. $WT^+ < WT^-$), (=) means that both algorithms perform equally well (i.e. $WT^+ = WT^-$). In bold, we report results in case H_0 (hypothesis of equal performance) is rejected at the 0.05 significance level

mainstream methods that do not consider autocorrelation (SVR, *k-NN*, M5', Base), by providing a remarkable reduction of the error for the most of the network datasets.

Table 2 presents the p-values of the Wilcoxon test comparing the errors obtained by NCLUS_I, NCLUS_RA and NCLUS_CI with the errors of the original CLUS algorithm. The analysis of these results reveals that NCLUS_I and NCLUS_RA give statistically better results than CLUS for three out of the five datasets (Movies1, Movies2 and Books) whereas NCLUS_CI results are significantly better than CLUS for two out of the five datasets (Movies2 and Books). For the other two datasets NCLUS results are generally better than those obtained by CLUS, but not significantly. This empirical study confirms our hypothesis that the explicit consideration of the non-stationary autocorrelation when building regression models from network data can increase the accuracy (decrease the error) of the obtained PCTs when autocorrelation is present in the data. Section 5.3.3 discusses some further characteristics of the learned PCTs.

5.3.2 Network data obtained from spatial datasets

Table 3 reports the average errors achieved by NCLUS, CLUS, CLUS*, ITL, SVR, k-NN and M5' on spatial datasets, as well as the errors of the Base model that always predicts the mean, on the spatial network datasets. Note that ITL builds model trees that consider spatial autocorrelation in a transductive network setting[8]. The best results are highlighted in bold.

As for the social network datasets, NCLUS errors only slightly depend on the adopted network autocorrelation measure. Indeed, the observed errors of PCTs learned by NCLUS do not change too much by varying this measure. However, once again, the best results are obtained with Moran's I. This result is not surprising, as Moran's

[8] Strictly speaking a comparison to the latter approach is not fair since model trees are recognized to be more accurate than regression trees. Moreover, ITL, according to the transductive learning settings, exploits both training and testing data during learning. We primarily use these results as a motivation for our further work and present the ITL improvements over the other algorithms in italic.

Table 3 Comparison of error made by different learning approaches on spatial network data

Method /Spatial Dataset		FF	NWE	FOIXA	GASD	MS	MF
NCLUS_I	$\alpha = 0.0$	**42.82**	**2.16**	2.53	0.18	2.47	5.44
NCLUS_I	$\alpha = 0.5$	56.55	2.48	**2.34**	0.17	**2.29**	5.81
NCLUS_RA	$\alpha = 0.0$	**42.82**	2.45	2.65	0.18	2.47	6.60
NCLUS_RA	$\alpha = 0.5$	53.27	2.46	2.66	0.17	2.35	5.92
NCLUS_CI	$\alpha = 0.0$	**42.82**	2.47	2.61	0.17	2.49	6.72
NCLUS_CI	$\alpha = 0.5$	52.79	2.45	2.66	0.16	2.35	5.93
CLUS		49.21	2.46	2.65	0.16	2.35	5.64
CLUS*		47.22	2.47	2.52	0.16	2.54	6.68
ITL		58.25	2.54	3.55	*0.14*	*1.92*	*3.52*
SVR		64.58	2.50	2.95	**0.14**	2.80	8.60
kNN		65.44	2.40	2.73	0.16	2.37	**4.56**
M5'		47.22	2.47	2.66	0.16	2.47	5.92
Base		63.66	2.50	2.93	0.20	3.23	8.58

The RMSEs (estimated by 10-fold CV) of the models obtained with NCLUS_I, NCLUS_RA, NCLUS_CI, CLUS, ITL, SVR, *k*-NN and M5', as well as the default Base model. For each network dataset, the best results are highlighted in bold. In the case ITL outperforms other approaches, we report values in italic. This because comparison to ITL is not fair. Results for NWE are multiplied by 10^3

I was specifically designed for modeling autocorrelation in spatial domains. Unlike for social network data, we have no clear evidence of the best accuracy been achieved with $\alpha = 0.5$. In general, considering autocorrelation is beneficial, but we cannot decide apriori how much the consideration of autocorrelation should influence the PCT construction (i.e., the value of $alpha$). In general, there is always a configuration of NCLUS that outperforms CLUS. Moreover, NCLUS outperforms CLUS*, except for the GASD dataset where they are comparable in performance.

Both, NCLUS and CLUS* are designed to improve (if possible) the accuracy of the CLUS PCTs by modifying/enhancing the heuristic (variance reduction for regression) used to evaluate each split in the process of tree construction. Whereas NCLUS accounts for autocorrelation that is often present in network data, CLUS* takes the spatial coordinates (usually presented in pairs (x, y) or (latitude, longitude)) from spatial datasets and considers them as response variables in addition to the actual response(s). This means that CLUS* aims at generating PCTs that will maximize the inter-cluster variance reduction of both the responses and the coordinates. Moreover, much higher importance is given to the spatial information than to the actual response, as they all are normalized at the beginning of the modeling process and equal importance is given the single target (response) and two coordinates x and y (as additional targets). This makes the predictions of the models more coherent in space than those of the CLUS models, mostly increases the accuracy of the models and shows some other characteristics of the models that will be discussed in Sect. 5.3.3.

However, in contrast to NCLUS models, CLUS* models cannot deal with non-stationary autocorrelation appropriately. In NCLUS, two different geographical regions that have the same distribution of attribute and target values including autocorrela-

Table 4 Statistical comparison of the performance of different NCLUS variants on spatial network data

Spatial/Method Dataset	NCLUS_I		NCLUS_RA		NCLUS_CI	
	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 0$	$\alpha = 0.5$
FF	(+)**0.01**	(−)0.56	(+)**0.01**	(−)0.56	(+)**0.01**	(−)0.56
NWE	(+)0.20	(−)0.96	(+)0.80	(=)1.00	(−)0.88	(+)0.80
FOIXA	(+)**0.01**	(+)**0.01**	(+)0.88	(+)0.88	(+)0.72	(−)0.80
GASD	(−)**0.01**	(−)0.28	(−)**0.01**	(−)0.28	(−)0.28	(=)1.00
MF	(−)0.88	(+)0.58	(−)0.88	(+)0.96	(−)0.88	(+)0.96
MS	(+)0.58	(−)0.88	(−)0.20	(−)0.58	(−)0.22	(−)0.58

The p-values of the Wilcoxon signed rank test comparing NCLUS and CLUS. (+) means that NCLUS is better than CLUS (i.e. $WT^+ > WT^-$), (−) means that CLUS is better than NCLUS (i.e. $WT^+ < WT^-$), (=) means that both algorithms perform equally well (i.e. $WT^+ = WT^-$). In bold, we report results in case H_0 (hypothesis of equal performance) is rejected at the 0.05 significance level

tion, can be covered by one leaf of the tree. In CLUS*, the data will need to be split into different regions due to the strong preference for spatial homogeneity. Moreover, CLUS* cannot handle different definitions of the regression problem that can arise from different definitions of the network, e.g., using different similarity measures. As for the social network datsets, NCLUS compares very well to mainstream methods that do not consider autocorrelation (SVR, *k-NN*, M5', Base), by providing a remarkable reduction of the error in most of spatial network datasets.

Table 4 presents the results of the Wilcoxon test when comparing NCLUS with the original CLUS algorithm in terms of the RMSE of the obtained PCTs. The errors obtained with NCLUS are statistically lower than those obtained with CLUS for the FF (using Global Moran's I/Relational Autocorrelation and $\alpha = 0$) and the FOIXA (using Global Moran's I) datasets and worse for the GASD dataset (using Global Moran's I/Relational Autocorrelation and $\alpha = 0$), at the significance level of 0.05. In the other case, there are at least two datasets on which each of the NCLUS algorithms wins over CLUS, but the difference is not statistically significant.

5.3.3 Properties of the models: size, autocorrelation of the errors and learning times

For completeness, we also include Table 5 that gives the mean and variance of the target variable calculated on the entire dataset and the correlation of the NCLUS_I, NCLUS_RA, NCLUS_CI, CLUS, CLUS*, SVR, k-NN and M5' predictions with the true target values on the test set. In most cases (7/11), NCLUS returns predictions which are more correlated with the true values than other competitive approaches. In the case of GASD and VideoL results are coherent with the errors, while in the case of FOIXA, k-NN shows slightly higher linear dependence between the predicted response value and the real one than NCLUS, although the error is smaller in NCLUS. For the Twitter dataset, NCLUS models result in negative correlation values due to over-pruning. Note that this is the smallest of all datasets considered.

Table 5 Mean and variance of the target variable calculated on the entire dataset and the correlation of the NCLUS_I, NCLUS_RA, NCLUS_CI, CLUS, CLUS*, SVR, *k*-NN and M5' model predictions with the true target values on the test set

Dataset	Mean	Variance	NCLUS_I		NCLUS_RA		NCLUS_CI		CLUS	CLUS*	SVR	*k*-NN	M5'
			α										
			0.0	0.5	0.0	0.5	0.0	0.5					
VideoL	359.20	$522 * 10^3$	0.34	0.42	0.34	0.42	0.34	0.42	0.35	–	0.29	0.52	0.44
MOVIES1	1993.63	53.23	0.85	0.86	0.85	0.86	0.85	0.85	0.59	–	0.56	0.27	0.52
MOVIES2	1985.12	385.77	0.88	0.91	0.88	0.91	0.87	0.83	0.67	–	0.46	0.53	0.34
BOOKS	5.66	6.40	0.12	0.12	0.12	0.12	0.12	0.12	0.12	–	0.12	0.00	0.12
TWITTER	5987.29	$173 * 10^6$	−0.32	−0.32	−0.32	−0.32	−0.32	0.28	−0.32	–	0.10	0.27	0.33
FF	12.95	4052.09	0.46	0.44	0.46	0.44	0.46	0.44	0.44	0.45	0.00	0.02	−0.12
NWE	10.00	9.00	0.14	0.08	0.10	0.10	0.09	0.10	0.08	0.06	0.10	0.02	0.11
FOIXA	0.93	8.58	0.34	0.34	0.33	0.33	0.33	0.33	0.26	0.34	0.28	0.37	0.30
GASD	9.76	1.74	0.46	0.57	0.46	0.57	0.57	0.59	0.59	0.59	0.70	0.68	0.58
MF	0.52	10.43	0.73	0.88	0.73	0.88	0.73	0.88	0.73	0.36	0.70	0.83	0.83
MS	0.05	3.24	0.87	0.81	0.83	0.80	0.80	0.81	0.70	0.45	0.77	0.79	0.77

Table 6 The average size of the PCTs learned by NCLUS, CLUS, CLUS* and M5'

Dataset/Method	NCLUS_I		NCLUS_RA		NCLUS_CI		CLUS	CLUS*	M5'
	α								
	0.0	0.5	0.0	0.5	0.0	0.5			
VideoL	3.0	6.9	3.0	4.9	3	6.9	6.9	–	11.0
MOVIES1	11.1	13.2	11.1	13.2	12.9	13.5	7.7	–	19.0
MOVIES2	7.9	10.8	7.9	10.8	7.9	11.9	7.5	–	11.0
BOOKS	1.0	1.0	1.0	1.0	1.0	1.0	4.8	–	9.0
TWITTER	1.7	1.7	1.7	1.7	1.0	1.7	2.2	–	4.0
FF	1.0	1.4	1.0	1.8	1.1	1.8	1.8	1.0	1.0
NWE	1.4	3.6	2.0	3.8	1.2	3.9	5.6	2.3	4.0
FOIXA	1.8	2.3	1.0	3.8	2.1	4.0	4.7	6.1	3.0
GASD	8.4	31.3	8.4	30.7	23.1	30.6	27.7	23.8	49.0
MF	1.0	4.4	1.0	4.2	3.4	4.1	5.1	19.7	6.0
MS	1.5	6.0	1.0	5.1	2.9	5.9	6.0	19.2	6.0

Table 6 shows the average size of the PCTs (number of leaves) learned by NCLUS, CLUS and CLUS*. In most cases, NCLUS learns smaller trees, whereas CLUS* learns larger trees than CLUS. This comes as a result of the consideration of the autocorrelation phenomenon in NCLUS models which, in most cases, makes the learned PCTs not only more accurate, but also smaller and consequently simpler to be displayed and interpreted. The predictions of the CLUS* PCTs are more coherent in space in comparison to CLUS PCTS, but differently from NCLUS, this happens at the price of increasing the size of the trees. While NCLUS can consider two different geographical regions that have the same distribution of attribute and target values (including autocorrelation) in one leaf of the tree, CLUS* will split these due to the emphasis on spatial homogeneity. This is the reason for the increase of the tree size.

Moreover, the PCTs learned by NCLUS by considering only the measures of network autocorrelation in the process of tree construction ($\alpha = 0$) are smaller than the models obtained with NCLUS using both autocorrelation and variance reduction in the tree construction ($\alpha = 0.5$). This comes as a result of the reduction in the number of relations/connections in the network with the introduction of additional splitting nodes, which then directly affects the calculation of the measures of network autocorrelation. This kind of situation is most notable in the models obtained using $NCLUS\_I$ and $NCLUS\_RA$, where the network autocorrelation is the only splitting criterion for the tree construction process. On the other hand, models that use only the CI index as a splitting criterion are less affected by this phenomenon as CI is only a function of the degree $Deg(v_i)$ of a node v.

In Table 7, we present autocorrelation of the prediction errors of the PCTs learned by NCLUS, CLUS, CLUS*, SVR, k-NN and M5'. Autocorrelation is computed by means of the Moran's I on the errors committed on the testing set. We analyze the obtained models in terms of this measure in order to show that PCTs learned by NCLUS can capture autocorrelation, when present in the network, and generate pre-

Table 7 Average autocorrelation of the prediction errors on the testing set, made by PCTs learned with NCLUS, CLUS, CLUS*, SVR, *k*-NN and M5'

Dataset/Method	NCLUS_I		NCLUS_RA		NCLUS_CI		CLUS	CLUS*	SVR	*k*-NN	M5'
	α										
	0.0	0.5	0.0	0.5	0.0	0.5					
VideoL	0.30	**0.22**	0.35	**0.22**	0.35	**0.22**	1.00	–	**0.22**	**0.22**	**0.22**
MOVIES1	−0.02	−0.02	−0.02	−0.02	**−0.01**	−0.02	−0.02	–	−0.02	−0.02	−0.02
MOVIES2	**−0.01**	−0.02	**−0.01**	**0.01**	**0.01**	−0.02	−0.02	–	**−0.01**	**−0.01**	**−0.01**
BOOKS	0.04	0.04	0.04	0.04	0.04	0.04	0.04	–	0.04	0.04	0.04
TWITTER	−0.50	0.50	−0.50	0.50	−0.50	**0.35**	0.76	–	−0.50	0.50	0.50
FF	**−0.02**	**−0.02**	**−0.02**	**−0.02**	**−0.02**	**−0.02**	1.00	**−0.02**	**−0.02**	**−0.02**	0.98
NWE	**0.00**	−0.01	−0.03	−0.02	−0.03	−0.02	0.84	−0.01	−0.01	−0.02	−0.01
FOIXA	**−0.02**	**−0.02**	**−0.02**	**−0.02**	**−0.02**	**−0.02**	0.96	**−0.02**	−0.03	−0.03	−0.06
GASD	0.19	0.19	0.11	0.19	0.07	0.05	1.00	0.08	**0.01**	0.03	0.37
MF	**−0.01**	0.15	**0.01**	0.07	0.08	0.06	0.88	0.15	**−0.01**	**0.01**	0.14
MS	0.13	0.24	0.03	0.055	**0.01**	0.04	0.66	0.13	**−0.01**	**−0.01**	0.34

For each dataset, the best results (the smallest in absolute value) are given in bold

dictions that exhibit small (absolute) autocorrelation in the errors. The analysis of the results reveals that NCLUS handles autocorrelation better than CLUS. In fact, coherently with the analysis reported in Sect. 4.5, NCLUS is able to correctly remove the effect of autocorrelation when making predictions. Thus, it is able to obtain network-consistent predictions. This analysis also reveals that CLUS* is able to capture autocorrelation better than CLUS, but worse than NCLUS. This is expected according to the differences between NCLUS and CLUS*, already discussed in this Section. Moreover, as expected, autocorrelation on the errors is often lower when $\alpha = 0$.

Table 8 reports the average learning times for NCLUS, CLUS, CLUS*, ITL, SVR, *k*-NN and M5' models. Results for CLUS* and ITL are available only for the spatial datasets. The shortest learning times are obtained by using the CLUS algorithm. The learning times for CLUS* are similar (slightly larger) to the times of CLUS, as in this configuration CLUS is run by considering the spatial coordinates as responses, while the time complexity of the PCT induction remains the same. The learning times for NCLUS are much longer than the learning times for CLUS because the consideration of autocorrelation introduces additional computations and increases the complexity of building a PCT. This is coherent with the time complexity analysis reported in Sect. 4.4. The learning times for ITL are significantly longer than the times of CLUS, CLUS* and NCLUS because of its iterative co-training implementation.

5.3.4 NCLUS for multi-target regression

In this section, we investigate the capability of NCLUS to adequately combine the effects of autocorrelation over several response variables when solving multi-target regression tasks. The results, presented in Table 9, demonstrate that there is no statistically significant difference between the accuracy of the multi-target PCTs and

Table 8 The learning times (seconds) of the NCLUS_I, CLUS, CLUS*, ITL, SVR, *k*-NN and M5'

Dataset/Method	NCLUS_I		CLUS	CLUS*	ITL	SVR	*k*-NN	M5'
	$\alpha = 0$	$\alpha = 0.5$						
VideoL	8.66	7.71	0.04	–	–	0.03	0.03	0.07
MOVIES1	130.21	141.17	0.07	–	–	0.04	0.04	1.02
MOVIES2	45.79	48.89	0.07	–	–	0.04	0.08	0.09
BOOKS	0.07	0.09	0.09	–	–	0.02	0.04	0.06
TWITTER	0.23	0.97	0.01	–	–	0.05	0.01	0.01
FF	1.50	1.03	0.04	0.04	422.10	1.03	0.45	1.23
NWE	2.31	1.62	0.11	0.11	857.30	1.22	1.02	1.94
FOIXA	0.69	0.49	0.02	0.02	162.90	0.49	0.56	0.88
GASD	27.86	20.79	0.04	0.04	30462.50	30.45	20.54	20.43
MF	2.68	1.39	0.04	0.03	593.60	1.00	2.00	2.25
MS	3.39	1.41	0.04	0.03	528.20	0.59	2.04	3.55

Table 9 The *p*-values of the Wilcoxon tests comparing the average RMSE of the single and multi-target PCTs learned by NCLUS

Dataset /Method	NCLUS_I		NCLUS_RA		NCLUS_CI	
	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 0$	$\alpha = 0.5$
MF	(=)1.00	(−)0.24	(−)0.96	(−)0.88	(+)0.51	(−)0.80
MS	(−)0.11	(−)0.51	(−)0.88	(−)0.65	(+)0.39	(−)0.33

(−) means that Single-Target PCTs are more accurate than Multi-Target PCTs; (+) means that Multi-Target PCTs are more accurate than Single-Target PCTs; (=) means that they perform equally well

the accuracy of the corresponding single-target PCT. This behavior is observed independently on the autocorrelation measure (NCLUS_I, NCLUS_RA and NCLUS_CI) used to learn the PCTs. In any case, we observe that the size of a single Multi-Target PCT is always significantly lower than the combined (by sum) sizes of the two single-target trees (see Table 10). This means that the multi-target PCTs learned by NCLUS adequately combine the effect of autocorrelation on several response variables by resulting in a predictive model that is accurate enough and simpler to be interpreted than several distinct trees. Moreover, the learning time spent to construct a multi-target PCT is lower than the learning times spent to learn several distinct single-target PCTs (see Table 11).

6 Conclusions

In this paper, we address the task of network regression. This is an important task as demonstrated by the presented related work and the use of real world datasets. While many approaches for network classification exist, there are very few approaches to the network regression task.

Table 10 The average sizes of the multi-target PCT learned by NCLUS compared to the size of single-target PCTs

Dataset/Method	NCLUS_I		NCLUS_RA		NCLUS_CI	
	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 0$	$\alpha = 0.5$
MS-MF	1.0	6.4	1.5	6.3	4.8	4.8
MF	1.0	4.2	1.0	4.2	3.4	4.1
MS	1.4	6.1	1.0	5.1	2.9	5.9

Table 11 The average learning time (seconds) of constructing a multi-target PCT and two single-target PCTs by NCLUS_I

Dataset/Method	NCLUS_I	
	$\alpha = 0$	$\alpha = 0.5$
MS-MF	4.73	2.7
MF	2.68	1.39
MS	3.39	1.41

The network setting that we address uses both the descriptive information (node attributes) and the network structure during training and uses only the descriptive information in the testing phase, where the network structure around the (new) testing instances may be unknown. This is quite different from existing approaches to network classification and regression, where the descriptive information used during both training and testing phase is typically closely related to the network structure and the connections between the nodes.

In this setting, we develop a data mining method that explicitly considers autocorrelation when building regression models from network data. The resulting models adapt to local properties of the data, providing, at the same time, smoothed predictions. The novelty of our approach is that, due to the generality of PCTs, it can work for different predictive modeling tasks, including regression and Multi-Target regression, as well as some clustering tasks.

We use well known measures of (spatial and relational) autocorrelation, since we deal with a range of different data networks. The heuristic we use in the construction of PCTs is a weighted combination of variance reduction (related to predictive performance) and autocorrelation of the response variable(s). Our approach can consider different sizes of neighborhoods (bandwidth) and different weighting schemes (degrees of smoothing) when calculating autocorrelation. We identify suitable combinations of autocorrelation metrics and weighting schemes and automatically determine the appropriate bandwidth.

We evaluate our approach on an extensive set of real world problems of network regression, coming from the areas of spatial and social networks. Empirical results show that the proposed extension of PCTs (NCLUS) performs better than both the original PCTs, which completely disregard network information, and PCTs that capture local spatial network regularities (CLUS*), but do not take autocorrelation into account and compares very well to mainstream methods that do not consider autocorrelation (SVR, *k-NN* and M5').

Our approach performs better along several dimensions. First, it is better in terms of predictive performance (as measured by RMSE, estimated by cross-validation). Second, autocorrelation of the errors made by our approach is smaller. Finally, the models produced by our approach are (on average) smaller.

Several directions for further work remain to be explored. The automated determination of the parameter α that sets the relative importance of variance reduction and autocorrelation during tree construction deserves immediate attention. In a similar fashion, one might consider selecting an appropriate autocorrelation measure. Moreover, it would be interesting to define novel autocorrelation measures that take indirect connections into account when computing autocorrelation. Finally, while our approach completely ignores the network information in the testing phase, we would like to explore developments in the direction of using this information if and when available.

Acknowledgments This work is in partial fulfillment of the research objectives of the project "EMP3: Efficiency Monitoring of Photovoltaic Power Plants" funded by Fondazione Cassa di Risparmio di Puglia and the PRIN 2009 project "Learning Techniques in Relational Domains and Their Applications" funded by the Italian Ministry of University and Research (MIUR). Džeroski is supported by the Slovenian Research Agency (Grants P2-0103, J2-0734, and J2-2285), the European Commission (Grants HEALTH-2008-223451, ICT-2010-266722, and ICT-2011-287713). He is also supported by Operation no. OP13.1.1.2.02.0005 financed by the European Regional Development Fund (85 %) and the Slovenian Ministry of Higher Education, Science and Technology (15 %). Stojanova is supported by the Slovenian Research Agency (Grant J2-2285). Thanks to Rich Davies for generating the MovieLens Dataset.

References

Aha D, Kibler D (1991) Instance-based learning algorithms, Machine Learning Journal, vol 6. Springer, Berlin

Angin P, Neville J (2008) A shrinkage approach for modeling non-stationary relational autocorrelation. In: Proceedings of 8th IEEE International Conference on Data Mining, IEEE Computer Society, pp 707–712

Antulov-Fantulin N, Bošnjak M, Žnidaršić M, Grčar M, Morzy M, Šmuc T (2011) Discovery challenge overview. In: ECML-PKDD 2011 Discovery Challenge Workshop, Springer, pp 7–20

Appice A, Ceci M, Malerba D (2009) An iterative learning algorithm for within-network regression in the transductive setting. In: Proceedings of 12th International Conference on Discovery Science, Springer, pp 36–50

Arthur G (2008) A history of the concept of spatial autocorrelation: A geographer's perspective. Geogr Anal 40(3):297–309

Basak D, Pal S, Patranabis D (2007) Support vector regression. Neural Inf Process Lett Rev 11(10):203–224

Bilgic M, Getoor L (2008) Effective label acquisition for collective classification. In: Proceedings of 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp 43–51

Blockeel H, De Raedt L, Ramon J (1998) Top-down induction of clustering trees. In: Proceedings of 15th International Conference on Machine Learning, Morgan Kaufmann, pp 55–63

Breiman L, Friedman J, Olshen R, Stone J (1984) Classification and regression trees. Wadsworth & Brooks, Monterey

Brent R (1973) Algorithms for minimization without derivatives. Prentice-Hall, Englewood Cliffs

Chuhay R (2010) Marketing via friends: strategic diffusion of information in social networks with homophily. Tech. Rep. 2010.118, Fondazione Eni Enrico Mattei

Cooper GF (1990) The computational complexity of probabilistic inference using bayesian belief networks (research note). Artif Intell 42:393–405

Cortez P, Morais A (2007) A data mining approach to predict forest fires using meteorological data. In: Proceedings of 13th Portuguese Conference on Artificial Intelligence, APPIA, pp 512–523

Cressie N (1993) Statistics for spatial data, 1st edn. Wiley, New York
Debeljak M, Trajanov A, Stojanova D, Leprince F, Džeroski S (2012) Using relational decision trees to model out-crossing rates in a multi-field setting. Ecol Modell. doi:10.1016/j.ecolmodel.2012.04.015
Demšar D, Debeljak M, Lavigne C, Džeroski S (2005) Modelling pollen dispersal of genetically modified oilseed rape within the field. In: Abstracts of the 90th ESA Annual Meeting, The Ecological Society of America, p 152
Dubin RA (1998) Spatial autocorrelation: a primer. J Hous Econ 7:304–327
Džeroski S, Gjorgjioski V, Slavkov I, Struyf J (2007) Analysis of time series data with predictive clustering trees. In: Proceedings of 5th International Workshop on Knowledge Discovery in Inductive Databases, Springer, pp 63–80
Engle RF (1982) Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom infation. Econometrica 50:987–1007
Epperson B (2000) Spatial and space-time correlations in ecological models. Ecol Model 132:63–76
Fotheringham AS, Brunsdon C, Charlton M (2002) Geographically weighted regression: the analysis of spatially varying relationships. Wiley, New York
Gallagher B, Tong H, Eliassi-Rad T, Faloutsos C (2008) Using ghost edges for classification in sparsely labeled networks. In: Proceedings of 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp 256–264
Ghimire J, Mani M, Crespi N (2008) A novel node connectivity index for wireless adhoc networks. Tech. rep., Télécom & Management SudParis
Glotsos D, Tohka J, Soukka J, Ruotsalainen U (2004) A new approach to robust clustering by density estimation in an autocorrelation derived feature space. In: Proc. 6th Nordic Symposium on Signal Processing, IEEE Computer Society, pp 296 – 299
Gora G, Wojna A (2002) RIONA: A classifier combining rule induction and k-NN method with automated selection of optimal neighbourhood. In: Proceedings of 13th European Conference on Machine Learning, Springer, pp 111–123
Grcar M, Lavrac N (2011) A methodology for mining document-enriched heterogeneous information networks. In: Proceedings of 14 International Conference on Discovery Science, vol 6926, Springer, pp 107–121
Griffith D (2003) Spatial autocorrelation and spatial filtering: gaining understanding through theory and scientific visualization. Advances in spatial science. Springer, Berlin
Hasan MA, Chaoji V, Salem S, Zaki M (2006) Link prediction using supervised learning. In: Proceedings of SDM Workshop on Link Analysis, Counterterrorism and Security, SDM
Jahani S, Bagherpour M (2011) A clustering algorithm for mobile ad hoc networks based on spatial autocorrelation. In: International Symposium on Computer Networks and Distributed Systems, IEEE Computer Society, pp 136 –141
Jensen D, Neville J, Gallagher B (2004) Why collective inference improves relational classification. In: Proceedings of 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp 593–598
Jin F (2010) Exploring spatial dependence: starting from the Moran's I and the APLE statistics. In: 20th Annual Meetings of the Midwest Econometrics Group
Kwak H, Lee C, Park H, Moon S (2010) What is twitter, a social network or a news media? In: Proceedings of 19th International Conference on World Wide Web, ACM, pp 591–600
Legendre P (1993) Spatial autocorrelation: trouble or new paradigm?. Ecology 74(6):1659–1673
LeSage JH, Pace K (2001) Spatial dependence in data mining. In: Grossman R, Kamath C, Kegelmeyer P, Kumar V, Namburu R (eds) Data mining for scientific and engineering applications. Kluwer Academic, Norwell pp 439–460
Li H, Calder CA, Cressie N (2007) Beyond Moran's I: testing for spatial dependence based on the spatial autoregressive model. Geogr Anal 39(4):357–375
Li X, Kang H, Cao J (2007b) Coordinated workload scheduling in hierarchical sensor networks for data fusion applications. In: MASS, ACM, pp 1–9
Macskassy S, Provost F (2007) Classification in networked data: a toolkit and a univariate case study. Mach Learn 8:935–983
Macskassy SA (2007) Improving learning in networked data by combining explicit and mined links. In: Proceedings of 22nd International Conference on Artificial Intelligence, AAAI Press, pp 590–595
McPherson M, Smith-Lovin L, Cook J (2001) Birds of a feather: homophily in social networks. Annu Rev Sociol 27:415–444

Mehta M, Agrawal R, Rissanen J (1996) SLIQ: A fast scalable classifier for data mining. In: Proceedings of 5th International Conference on Extending Database Technology, Springer, pp 18–32
Michalski RS, Stepp RE (1983) Learning from observation: conceptual clustering. In: Michalski RS, Carbonell JG, Mitchell TM (eds) Machine learning: an artificial intelligence approach. Tioga, Palo Alto pp 331–364
Neville J, Jensen D (2007) Relational dependency networks. J Mach Learn Res 8:653–692
Neville J, Simsek O, Jensen D (2004) Autocorrelation and relational learning: challenges and opportunities. In: Proceedings of Workshop Statistical Relational Learning, AAAI Press, pp 290–299
Newman MEJ, Watts DJ (2006) The structure and dynamics of networks. Princeton University Press, Princeton
Orkin M, Drogin R (1990) Vital statistics. McGraw Hill, New York
Pace P, Barry R (1997) Quick computation of regression with a spatially autoregressive dependent variable. Geogr Anal 29(3):232–247
Popescul A, Ungar LH (2003) Statistical relational learning for link prediction. In: Proceedings of IJCAI Workshop on Learning Statistical Models from Relational Data, IJCAI, pp 172–182
Quinlan RJ (1993) C4.5: programs for machine learning. Morgan Kauffmann, San Francisco
Rahmani H, Blockeel H, Bender A (2010) Predicting the functions of proteins in protein-protein interaction networks from global information. J Mach Learn Res 8:82–97
Randic M (1998) On characterization of molecular attributes. Acta Chim Slovenica 45:239–252
Sen P, Namata G, Bilgic M, Getoor L, Gallagher B, Eliassi-Rad T (2008) Collective classification in network data. AI Mag 29:3–93106
Steinhaeuser K, Chawla NV, Ganguly AR (2011) Complex networks as a unified framework for descriptive analysis and predictive modeling in climate science. Stat Anal Data Min 4(5):497–511
Stojanova D, Ceci M, Appice A, Dzeroski S (2011a) Network regression with predictive clustering trees. In: ECML/PKDD (3), Springer, pp 333–348
Stojanova D, Ceci M, Appice A, Malerba D, Džeroski S (2011b) Global and local spatial autocorrelation in predictive clustering trees. In: Proceedings of 14 International Conference on Discovery Science, vol 6926, Springer, pp 307–322
Vapnik V (1998) Statistical learning theory. Wiley, New York
Wang Y, Witten I (1997) Induction of model trees for predicting continuous classes. In: Proceedings of Poster Papers of the European Conference on Machine Learning, Faculty of Informatics and Statistics, University of Economics, Prague, pp 128–137
Weng J, Lim E, Jiang J, He Q (2010) Twitterrank: finding topic-sensitive influential twitterers. In: Proceedings of 3rd ACM International Conference on Web Search and Data Mining, ACM, pp 261–270
Witten I, Frank E (2005) Data mining: practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco
Zhu X, Ghahramani Z, Lafferty JD (2003) Semi-supervised learning using gaussian fields and harmonic functions. In: Proceedings of 20th International Conference on Machine Learning, AAAI Press, pp 912–919
Ziegler C, Mcnee S, Konstan J, Lausen G (2005) Improving recommendation lists through topic diversification. In: Proceedings of 14th International Conference on World Wide Web, ACM, pp 22–32