Data Mining and Constraints: An Overview

Valerio Grossi(✉), Dino Pedreschi, and Franco Turini

Department of Computer Science, University of Pisa,
Largo B. Pontecorvo, 3, 56127 Pisa, Italy
{vgrossi,pedre}@di.unipi.it, turini@unipi.it

Abstract. This paper provides an overview of the current state-of-the-art on using constraints in knowledge discovery and data mining. The use of constraints requires mechanisms for defining and evaluating them during the knowledge extraction process. We give a structured account of three main groups of constraints based on the specific context in which they are defined and used. The aim is to provide a complete view on constraints as a building block of data mining methods.

1 Introduction

Data mining extracts synthetic models from datasets. Data are represented by collections of records characterizing data with respect to several dimensions. The use of constraints may be useful in the data mining process in at least three ways: ***(i)*** *filtering* and *organizing* the dataset before applying data mining methods; ***(ii)*** *improving* the performance of data mining algorithms by reducing the search space and focusing the search itself; and ***(iii)*** *reasoning* on the results of the mining step for sharpening them and presenting a more refined view of the extracted models.

The integration of constraints in data mining tasks has rapidly emerged as a challenging topic for the research community. A large number of ad-hoc extensions of mining algorithms use constraints for improving the quality of their results. The use of constraints requires a way for defining and satisfying them during the knowledge extraction process. This point is crucial both for the quality of the extracted data mining models, and for the scalability of the entire process. On the one hand, an analyst can define the knowledge extraction phase where a constraint must be satisfied. On the other hand, an optimizer is required to understand where a constraint must be satisfied inside the process flow, in an automatic way. Moreover, mining algorithms must be rewritten for satisfying constraints directly into model extraction.

The amount of data in our world has been exploding. This chapter ends offering the user a glimpse at the future by considering the emerging phenomenon of *big data*. With big data traditional analysis tools cannot be used because of the massive volume of data gathered by automated collection tools, there are already promising line researches addressing this issue.

Furthermore, this chapter represents a solid scientific basis for several advanced techniques developed inside the ICON project and outlined in this book.

C. Bessiere et al. (Eds.): Data Mining and Constraint Programming, LNAI 10101, pp. 25–48, 2016.
DOI: 10.1007/978-3-319-50137-6_2

For example the reader can examine in depth the use of a constraint language for defining data mining tasks considering the Chapter "*Modeling Data Mining Problems in MiningZinc*", or study clustering problems via constraints optimization reading the Chapter "*Partition-Based Clustering using Constraints Optimization*".

For these aims, Sect. 2 provides an introduction to data mining and proposes several references useful to understand how the basic data mining concepts can be extended by using constraints. Section 3 reviews the use of constraints in data mining, introducing three different dimensions on which constraints can be classified. Finally, Sect. 4 draws some conclusions.

2 Data Mining

Today, data mining is both a technology that blends data analysis methods with sophisticated algorithms for processing large data sets, and an active research field that aims at developing new data analysis methods for novel forms of data. On the one hand, data mining tools are now part of mature data analysis systems and have been successfully applied to problems in various commercial and scientific domains. On the other hand, the increasing heterogeneity and complexity of new forms of data, such as those arriving from medicine, biology, the Web, Earth observation systems, call for new forms of patterns and models, together with new algorithms to discover such patterns and models efficiently.

Data mining is originally defined as the process of automatically discovering useful information in large data repositories. Traditionally, data mining is only a step of knowledge discovery in databases, the so-called KDD process for converting raw data into useful knowledge. The KDD process consists of a series of transformation steps: *data preprocessing*, which transforms the raw source data into an appropriate form for the subsequent analysis. *Actual data mining*, which transforms the prepared data into patterns or models, and *postprocessing of mined results*, which assesses validity and usefulness of the extracted patterns and models, and presents interesting knowledge to the final users - business analysts, scientists, planners, etc. – by using appropriate visual metaphors or integrating knowledge into decision support systems.

The three most popular data mining techniques are *predictive modelling*, *cluster analysis* and *association analysis*. In predictive modelling (Sect. 2.1), the goal is to develop classification models capable of predicting the value of a class label (or target variable) as a function of other variables (explanatory variables); the model is learnt from historical observations, where the class label of each sample is known: once constructed, a classification model is used to predict the class label of new samples whose class is unknown, as in forecasting whether a patient has a given disease based on the results of medical tests.

In association analysis, also called pattern discovery, the goal is precisely to discover patterns that describe strong correlations among features in the data or associations among features that occur frequently in the data (see Sect. 2.3). Often, the discovered patterns are presented in the form of association rules:

useful applications of association analysis include market basket analysis, i.e. the task of finding items that are frequently purchased together, based on point-of-sale data collected at cash registers.

Finally, in cluster analysis (Sect. 2.2), the goal is to partition a data set into groups of closely related data in such a way that the observations belonging to the same group, or cluster, are similar to each other, while the observations belonging to different clusters are not. Clustering can be used, for instance, to find segments of customers with a similar purchasing behaviour or categories of documents pertaining to related topics.

2.1 Predictive Modelling or Classification

Classification is one of the most popular approaches for mining useful information. The aim is to predict the behavior of new elements (*classification phase*), given a set of past and already classified instances. The process of classifying new data begins from a set of classified elements, and tries to extract some regularities from them (*training phase*) [WFH11, TSK06, HK12]. The model employs a set of input data called *training set* where the class label for each instance is provided. The process of classifying new data starts from a training set, and tries to extract some regularities from them. Classification is an example of *supervised learning*.

Based on the way learners actually subdivide the above-mentioned phases, they are categorized into two classes, namely *eager learners* or *lazy learners*. For example, decision trees or rule-based learners are examples of *eager* approaches. In this category, most of the computing resources are spent to extract a model, but once a model has been built, classifying a new object is a rather fast process.

By contrast, lazy learners, such as *nearest-neighbour classifiers* do not require an explicit model building phase, but classifying a test example can be very expensive, since the element to classify must be compared with all the samples in the training set. In the following, we provide a short description of the most popular classifiers available in the literature.

Decision Trees. The model has the form of a tree, where each node contains a test on an attribute, each branch from a node corresponds to a possible outcome of the test, and each leaf contains a predicted class label [Mor82]. Decision tree induction often uses a greedy top-down approach which recursively replaces leaves by test nodes, starting from the root. The attribute associated to each node is chosen through the comparison of all the available attributes, and the selection of the best one is based on some heuristic measures. Several impurity measures are available in the literature [Qui86, Qui93, BFOS84]. Typically, the measures developed are based on the degree of impurity of the child nodes. The lower is the value, the more skewed is the class distribution. The extraction procedure continues until a termination condition is satisfied.

The Hunt's algorithm represented in Algorithm 1 is the basis of several popular decision tree learners including ID3 [Qui86], CART [BFOS84], C4.5 [Qui93, Qui96] and EC4.5 [Rug02]. The cited approaches assume that all training

Algorithm 1. The Hunt's algorithm - *DecisionTree(TS, A)*

Require: Training set *TS*, an attribute set *A*
Ensure: Decision tree
1: **if** *stoppingCondition(TS, A)* = *true* **then**
2: *leaf* ← *createLeaf(TS)* //given *TS* determines the class label to assign a leaf node
3: **return** *leaf*
4: **else**
5: *root* ← *createNode()*
6: *root.testCondition* ← *findBestSplit(TS, A)*
7: TS_i ← *splitData(root.testCondition)* //given the test condition splits *TS* in subsets
8: **for each** TS_i **do**
9: $root.child_i$ ← *DecisionTree*(TS_i, *A*)
10: **end for**
11: **end if**
12: **return** *root*

examples can be simultaneously stored in main memory, and thus have a limited number of examples from which they can learn. [LLS00] shows a comparison of complexity, training time and prediction accuracy of main memory classification algorithms, including decision trees. In several cases, training data can exceed the main memory capability. In order to avoid this limitation, disk-based decision tree learners, such as SLIQ [MAR96] and SPRINT [SAM96], assume the examples to be stored on disk, and are learned by repeatedly reading them in a sequence. More recently, new data structures and algorithms have been defined to tackle the classification problem in stream environments, also using decision trees [GT12, GS11].

Bayesian Approaches. In many situations, the relationship between the attributes and the class variable cannot be deterministic. This situation typically occurs in the presence of noisy data, or when external factors affecting classification, not included in our analysis, arises. Based on Bayes theorem, Bayesian classifiers are robust to isolate noisy points and irrelevant attributes.

A popular approach of Bayesian classification is naïve Bayes. This kind of classifier estimates the class-conditional probability, by assuming that the attributes are conditionally independent. To classify a record, the algorithm computes the posterior probability of a class value using Bayes theorem, and returns the class that maximizes this probability value. The way of computing class-conditional distribution varies in the presence of categorical or continuous attributes. In the first case, the conditional probability is estimated using the fraction of training samples with a specific class label considering an attribute value. By contrast, continuous attributes must be discretized, or a Gaussian distribution is typically chosen to compute the class-conditional probability.

Detailed discussions on Bayesian classifiers can be found in [DH73, Mic97, WK91]. An analysis of the accuracy of naïve Bayes classifiers without class

Algorithm 2. The k-nearest neighbour algorithm

Require: Training set TS, the number of nearest neighbour k
Ensure: Set of k nearest neighbours
1: **for each** test example $z = (x', y')$ do **do**
2: $Distance(x', x) \leftarrow$ compute the distance between z and every training element $(x, y) \in TS$
3: $TS_s \leftarrow$ Select the k closest training example to z
4: $class \leftarrow FindClass(TS_s)$
5: **return** $class$
6: **end for**

conditional independence hypothesis is available in [DP96], while [Jen96] provides a first overview of Bayesian networks.

Nearest Neighbour. This kind of classifier belongs to the family of lazy learners. In this case, every training example is viewed as a point in a multidimensional space, defined on the number of the available attributes.

As shown in Algorithm 2, given an element to classify, the call label is chosen based on the label of element neighbours selected by a proximity measure. In this case, specific training instances are employed to provide a prediction, without providing any model derived from data. Every training example is viewed as a point in a multidimensional space, defined on the number of the available attributes. In real applications only k points, that are closest to the element to classify are selected to decide the class label to return. The crucial aspect is to select the measures of proximity, that similarly to clustering are based on attribute types and special issues to solve. Due to its nature these models are rather sensible to noisy data and the prediction accuracy is highly influenced by the data preprocessing step and proximity measure.

With respect to decision trees, nearest-neighbor classifier provides a more flexible model representation. It produces arbitrarily-shaped boundaries, while decision trees are typically constrained to rectilinear decision boundaries [TSK06, HK12].

Support Vector Machine. This kind of approaches has its root in statistical learning theory. They have been successfully employed in many real applications, including handwritten digit recognition, and text categorization among others.

The main idea of this method is representing the decision boundary using a subset of training examples, known as support vectors. A support vector machine constructs a hyperplane (or set of hyperplanes) in a multi-dimensional space, which can be used for classification, regression, or other tasks. Essentially, given a set of possible hyperplanes (implicitly defined in the data), the classifier selects one hyperplane for representing its decision boundary, based on how well they are expected to perform on test examples. A support vector approach is typically described as linear or non-linear. The former involves a linear decision boundary to split the training objects into respective classes [ABR64]. Non-linear models

try to compute a boundary for separating objects that cannot be represented by a linear model [BGV92]. The trick is to transform the data from its original space into a new space that can be divided by a linear bound. In the literature several approaches are available for learning a support vector model [CV95, Bur98, SC08].

2.2 Clustering

Clustering is the process of partitioning a set of data objects into subsets without any supervisory information such as data labels. Each subset is a cluster, such that objects in a cluster are similar to one another, yet dissimilar to objects in other clusters. The set of clusters resulting from a cluster analysis can be referred to as a clustering [WFH11, TSK06, HK12]. Clustering can lead to the discovery of previously unknown groups within the data. Examples of data objects include database records, graph nodes, a set of features describing individuals or images. Because there is no a priori knowledge about the class labels, clustering is also called unsupervised learning. Cluster analysis is used in a wide range of applications such as: business intelligence, image pattern recognition, web analysis, or biology.

The following general aspects are orthogonal characteristics in which clustering methods can be compared:

- **the partitioning criteria**: all the clusters are at the same level *vs.* partitioning data objects hierarchically, where clusters can be formed at different semantic levels.
- **separation of clusters**: methods partitioning data objects into mutually exclusive clusters *vs.* a data object may belong to more than one cluster.
- **similarity measure**: similarity measures play a fundamental role in the design of clustering methods. Some methods determine the similarity between two objects by the distance between them *vs.* the similarity may be defined by *connectivity* based on density or contiguity.
- **clustering space**: the entire given data space *vs.* subspace clustering.

The literature proposes several ways to compute and represent a cluster. The partition method is based on prototypes and is one of the most widely studied and applied approaches. In this case, every cluster is selected and represented by a prototype called *centroid* (e.g. *K-means* and *K-medoid*). Prototype-based techniques tend to consider the region only based on a distance value from a center. This approach typically provides clusters having globular shapes. Hierarchical-clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. Also this kind of clustering is typically based on distance measures, but in this case, we permit clusters to have subclusters thus forming a tree. Each cluster i.e. a node in the tree, is the union of its subclusters, and the root of the tree is the cluster containing all the objects. The class of approaches for hierarchical clustering can be found under the agglomerative hierarchical clustering. BIRCH [ZRL96] is a famous example of hierarchical clustering algorithm.

Algorithm 3. The k-means algorithm

Require: Set of points P
Ensure: Set of k clusters
1: **repeat**
2: Form k clusters by assigning each point $p_i \in P$ to the closest centroid
3: *centroids* ← Recompute the centroid of each cluster
4: **until** *centroids* do not change

Density-based approaches work also with non-globular regions and they are designed for discovering dense areas surrounded by areas with low density (typically formed by noise or outliers). In this context a cluster consists of all density-connected objects, which can form a cluster of an arbitrary shape. DBSCAN [EKSX96] and its generalization OPTICS [ABKS99] are the most popular density based clustering methods. In several situations spectral and/or graph-based clustering are proposed for solving problems when the available information is encoded as a graph. If the data is represented as a graph, where the nodes are objects and the links represent connections among objects, then a cluster should be redefined as a connected component, i.e. a group of objects that are connected to one another, but that have no connection to objects outside the group. An important example of graph-based clusters are contiguity-based clusters, where two objects are connected only if they are within a specified distance of each other. This implies that each object in a contiguity-based cluster is closer to some other object in the cluster than to any point in a different cluster.

Finally, Fig. 1, taken from [HK12], summarizes the main characteristics related to the different clustering approaches considering the three main clustering methods proposed above. For each method, the figure highlights the specific features and the most well-known and basic algorithms widely studied in the literature. Finally, Fig. 1, taken from [HK12], summarizes the main characteristics related to the different clustering approaches considering the three main clustering methods proposed above. For each method, the figure highlights the specific features and the most well-known and basic algorithms widely studied in the literature.

Method	Specific Features	Algorithms
Partitioning methods	Distance based Discover mutual clusters of spherical shape Prototyped-based (mean or medoid) to represent centroid	K-means K-medoids
Hierarchical methods	Hierarchical decomposition May incorporate other techniques (e.g. microclustering) Cannot correct erroneous splits (or merges)	BIRCH
Density-based methods	Find arbitrary shaped clusters Based on concept of dense regions May filter out outliers	DBSCAN OPTICS

Fig. 1. Overview of clustering methods.

2.3 Pattern Discovery

Pattern analysis methods are fundamental in many application domains including market basket analysis, medicine, bioinformatics, web mining, network detection, DNA research. Unlike in predictive models, in pattern discovery the objective is to discover all patterns of interest. Here, we briefly recall the basic methods of pattern mining, including *frequent itemsets mining* (FIM), *association rule mining* (ARM) and *sequential patterns mining* (SPM). See [ZZ02, HCXY07, Sha09] for past surveys on ARM, and [ME10, CTG12] for surveys on SPM.

Let $I = \{i_1, \ldots, i_n\}$ be a set of distinct literals, called items. An itemset X is a subset of I. An itemset X has a support, $supp(X)$, in a transactional database D if $s\%$ of the transactions contains the itemset X in D. Given a user-defined minimum support $\overline{s}$, an itemset X such that $supp(X) \geq \overline{s}$ is called frequent itemset. The FIM problem can be stated as follows: given a transaction database D and a minimum support threshold $\overline{s}$, find all the frequent itemsets from the set of transactions w.r.t. $\overline{s}$.

A natural derivation of frequent itemsets is called association rule (AR), expressing an association between two itemsets. Given X and Y two itemsets, with $X \cap Y = \emptyset$, an AR is an expression of the form $X \Rightarrow Y$. X is called the body or antecedent, and Y is called the head or consequent of the rule. The support of an AR $X \Rightarrow Y$ is $supp(X \Rightarrow Y) = supp(X \cup Y)$. The confidence of an AR is $conf(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$. Given a transaction database D, a minimum support threshold, $\overline{s}$, and a minimum confidence threshold, $\overline{c}$, the ARM problem is to find all the ARs from the set of transactions w.r.t. $\overline{s}$ and $\overline{c}$.

Finally, the concept of sequential pattern is introduced to capture typical behaviors over time, i.e. behaviors sufficiently repeated by individuals to be relevant for the decision maker. A sequence $S = < X_1 \ldots X_n >$ is an ordered list of itemsets. We say that S is a subsequence of another sequence $V = < Y_1 \ldots Y_m >$ with $n \leq m$, if there exist integers $1 \leq i_1 < \cdots < i_n \leq m$ such that $X_1 \subseteq Y_{i1}, \ldots, X_n \subseteq Y_{in}$. We denote with X_i.time the timestamp of the itemset X_i and with $supp(S)$ the support of S, i.e. the number of tuples containing the sequence S. Given a sequence database and a minimum support threshold $\overline{s}$, the SPM problem is to find all the sequences from the set of transactions w.r.t. $\overline{\sigma}$. Sequential patterns are not the only form of patterns that can be mined. Consider for example the huge literature for gene mining [EZ13].

Different algorithms for FIM have been proposed in the literature [AS94, HPY00, SON95, Toi96, ZPOL97]. The most popular algorithm is Apriori [AS94]. The approach is outlined in Algorithm 4. It is based on a level-wise search process that makes multiple passes over the data. Initially, it computes the frequent itemsets of size 1. The core of the algorithm is then a cycle of passes each of them composed of two main phases: the candidate generation and the support counting. In the former phase, the set of all frequent k-itemsets, L_k, found in the pass k, is used to generate the candidate itemsets C_{k+1}. In the latter, data is scanned to determine the support of candidates. After the support counting, unfrequent itemsets are dropped, according to the *downward closure property*. Another algo-

Algorithm 4. The Apriori algorithm

Require: Set of transaction T
Ensure: Frequent itemsets
1: $k \leftarrow 1$
2: $F_k \leftarrow$ Find all frequent 1-itemsets
3: **repeat**
4: $k \leftarrow k + 1$
5: **for each** transaction $t \in T$ **do**
6: Identify all candidates that belongs to t
7: Compute support counting for each candidate C_t
8: **end for**
9: $F_k \leftarrow$ Extract the frequent k-itemsets
10: **until** $F_k = \emptyset$
11: **return** $\bigcup F_k$

rithm is the FP-Growth. It allows to reduce the number of transactions to be processed at each iteration via a *divide et impera* strategy [HPY00]. Basically, it divides the search space on a prefix base. After the first scan, the original problem can be divided into $|I|$ sub-problems, where I is the set of frequent singletons. Other algorithms based on the splitting of the input data into smaller datasets, are eclat [ZPOL97] and partition [SON95].

Sequential pattern mining methods can be classified into three classes: Apriori-based with an horizontal formatting methods; Apriori-based with a vertical formatting methods; projection-based pattern growth methods. The first class includes the GSP algorithm [SA96] and its derivations. The second class includes SPADE [Zak01]. The third class is based on the SPAM [AFGY02] and PrefixSpan algorithms [PHMA+04]. In particular, the latter works by means of a divide-and-conquer strategy with a single scan on the entire dataset. Each sequential pattern is treated as a prefix and mined recursively over the corresponding projected database.

Recently, mining frequent structural patterns from graph databases, e.g. web logs, citation networks, and social networks has become an important research problem with broad applications. Several efficient algorithms were proposed in the literature [WWZ+05, IWM00, YH02], ranging from mining graph patterns, with and without constraints, to mining closed graph patterns.

3 Using Constraints in Data Mining

The integration of constraints in data mining has rapidly emerged as a challenging topic for the research community. Many ad-hoc extensions of mining algorithms that use constraints for improving the quality of their results have been proposed for the different methods introduced along the Sect. 2. The definition and the integration of constraints allows the user to specify additional information on input data as well as requirements and expected properties of data mining models in output in a declarative way. For example, the extraction of association rules typically leads to a large quantity of useless rules.

An approach that extracts the rules by specifying the analyst's needs can speed up both the domain experts evaluation of the extracted rules and the extraction algorithm itself.

The literature proposes several works on using constraints in data mining tasks. Currently, every mining task has its own way for classifying constraints. A full view that binds mining tasks to the the objects on which constraints are defined, is still missing. For this reason, one of the aims of this chapter is to provide a general framework where a constraint can be classified. In this perspective, this section provides a description about the dimensions on which constraints can be classified. This view is based on the main characteristics that every kind of constraint proposes in its specific mining context.

We introduce the use of constraints considering three dimensions based on the characteristics that every kind of constraint presents in its specific context:

1. **Object Constraints**: considers which **objects** the constraints are applied to, namely *data*, *models* and *measures*. This kind of constraints is presented in Sect. 3.1.
2. **Hard &Soft Constraints**: considers the **type** of constraints: *hard* and *soft* constraints. Section 3.2 introduces this kind of constraints.
3. **Phase-defined Constraints**: considers the **phases** of the knowledge extraction process, in which the constraints are used, namely *pre*, *mining* and *post*. Section 3.3 overviews this class of constraints.

Before starting analysing the dimension dealing with the **objects** constraints, it is worth noting that the dimensions proposed above are not complementary or mutually exclusive, but they represent different perspectives on which we can classify constraints for data mining.

3.1 Object Constraints

We start by analyzing the dimension dealing with the **objects** constraints are applied to. Constraints can be defined on *data*, on the *mining model* and on *measures*. In particular, Sect. 3.1.1 overviews the constraints on data (or items), while Sect. 3.1.2 overviews the ones on mining models. Finally, Sect. 3.1.3 introduces the constraints defined on measures.

3.1.1 Constraints on Data

Referred to the literature also as constraints on *items*, this kind of object constraint involves specific data attributes. Data constraints require a complete knowledge about the data attributes and properties in order to define constraints on specific data features. Furthermore, they can involve some forms of background knowledge directly. Examples of constraints on data include the *must and cannot-link* in a clustering problem, or *consider only the items having a price higher than a given threshold* for pattern mining.

If we consider the *classification* task the literature in this field has explored constraints among *instances and classes*, and among different *classes* themselves.

This is principally due to the fact that a classifier is extracted from a training set specifically conceived on the requirements of the classification task. [HPRZ02] introduces a constrained classification task, where each example is labeled with a set of constraints relating multiple classes. Every constraint specifies the relative order of two classes and the goal is to learn a classifier consistent with these constraints. As reported in [PF08], in many applications explicit constraints among the labels can be easily discovered. For example, in the context of hierarchical classification, the presence of one label in the hierarchy often implies also the presence of all its ancestors. [TJHA05] proposes a constrained *support vector machine* approach. In this work, the authors consider cases where the prediction is a structured object or consists of multiple dependent constrained variables. An interesting approach is proposed in [DMM08] in case of a lack of labeled instances. In this case, the knowledge base is a set of labeled features, and the authors propose a method for training probabilistic models with labeled features (constrained from domain knowledge) from unlabeled instances. Labeled features are employed directly to constrain the model predictions on unlabeled instances.

Data constraints for *clustering* involves the concept of *instance-level* constraints. Well-established approaches on using data constraints for clustering problems focused on the introduction of instance-level constraints [WCRS01, WC00]. In this case a domain expert defines constraints that bind a pair of instances in the same cluster or that avoid that a pair of instances will be assigned to the same cluster. ***(i)*** *must-link* constraints enforce two instances to be placed in the same cluster, while ***(ii)*** *cannot-link* constraints enforce two instances to be in different clusters. Several properties are related to instance-level constraints [DR06]. Must-link constraints are symmetric, reflexive and transitive. The latter property enables a system to infer additional must-link constraints. On the contrary, cannot-links do not have the transitive property. Since *must* and *cannot*-link are relevant for a large amounts of works in the literature, where several types of constraints based on groups of instances have been defined in [DR05, DR09, DR07, DDV13], Chap. 1 in [BDW08] reports a detailed definition of the properties on which they are based.

In *pattern mining*, data constraints are introduced to specify patterns that include (or not) specific items. For example, when mining association rules out of a weblog, one might be interested in only rules having sport pages in the consequent, and not having shopping pages in the antecedent. In the case of sequential patterns, one might be interested to patterns that first visit finance, and then sport or books [PHW07]. There are two principal ways to express data constraints for pattern mining: ***(i)*** by means of a *concept hierarchy* (i.e. multi-level constraints) and ***(ii)*** *weighted pattern mining* emerges when considering a different semantic significance of the items.

Multi-level constraints enables the generalization of items at bottom level to higher levels of the hierarchy before applying the mining algorithm [SA95]. Methods to integrate multi-level constraints into mining algorithms are introduced in [HF99], in which frequent itemsets are generated one level at a time

of the hierarchy. [SVA97] and [HLN99] can be seen as the first attempts to integrate multilevel mining directly into the Apriori. More recent works on generalized rule mining include [ST02] about exploiting the lattice of generalized itemsets, and [WH11], on using efficient data structures to retrieve item generalizations. [BCCG12] exploits schema constraints and the opportunistic confidence constraints to remove uninteresting rules.

Weighted pattern mining has been extensively proposed in *frequent itemset mining* and *association rule mining*, in discussing a new tree structure that is robust to database modifications [ATJ+12]; in pushing the weight constraint into pattern growth algorithms [YL05, TSWYng, YSRY12], or into level-wise methods [WYY00, TM03, LYC08]; in suggesting approximated weighted frequent pattern mining, as a fault tolerant factor [YR11].

3.1.2 Constraints on the Mining Model

This class of constraints defines specific requirements that an extracted model should satisfy. This kind of constraint does not involve background knowledge directly, but it requires a complete knowledge on the characteristics needed by the output model. For example, they include the extraction of association rules having a specific set of items in the body and in the head, or discovering clusters with a minimum number of elements.

Examples of model constraints for classification can be found in [NF07, NF10, NPS00]. [NPS00] proposes different kinds of constraints, related to the form of a decision tree, e.g. internal nodes should not have pure class distributions or rules about the class distribution. [NF10] defines a framework for determining which model constraints can be pushed into the pattern mining process, proposing an optimal classifier model. More precisely, [NF10] shows how several categories of constraints defined for frequent itemset mining, e.g. *monotonic*, *anti-monotonic* and *convertible*, can be applied in decision tree induction. It highlights the connection between constraints in pattern mining and constraints in decision tree extraction, developing a general framework for categorizing and managing decision tree mining constraints.

The algorithms K-means and K-medoid represent a basic approach for forcing clustering models to have specific properties [GMN+15]. In [BBD00, DBB08], the authors avoid empty clusters by adding k constraints to the clustering problem requiring that cluster h contains at least τ_h points. The solution proposed is equivalent to a minimum cost flow linear network optimization problem [Ber91]. Another approach for discovering balanced clusters can be found in [BG08, BG06]. In this case, the introduced constraint requires that the obtained clusters have a comparable size. The proposed method has three steps: ***(i)*** sampling; ***(ii)*** clustering of the sampled set; and ***(iii)*** populating and refining the clusters while satisfying the balancing constraints. Other methods for constraining the clustering approach to discover balanced clusters can be found in [SG03]. The authors propose the use of graph partition techniques or hierarchical approaches that encourage balanced results while progressively merging or splitting clusters [BK03, ZG03]. Many papers focus on metric learning driven

by constraints. Distance measure learning and clustering with constraints in K-means were both considered in [BBM04b], and the result was extended to a Hidden Markov random field formulation in [BBM04a].

Pattern-model constraints are related to the form, or the structure of the entire pattern, as well as to relations among items. For example, one might wish to find patterns that include first visit of a sport page, then a shopping page, and finally a finance page. In this context, we are searching for meta-rules that are useful to specify the syntactic form of the patterns [FH95]. These constraints can be specified using either high-level user interfaces or declarative data mining query languages. Here, we briefly review the usage of regular expressions (RE) in sequential pattern mining. They are based on the typical RE operators, such as disjunction and Kleene closure, to constrain the set of items. Then, we deal with relaxation of constraints. There are several algorithms supporting RE constraints. SPIRIT [GRS99] is based on an evolution of the GSP algorithm. RE-Hackle represents RE by means of a tree structure [CMB03]. Prefix-growth extends the prefix-span approach with several kinds of constraints, among which RE are included [PHW07].

3.1.3 Constraints on Measures

Measures, e.g. *entropy* for classification, *support* and *confidence* for frequent itemsets and *euclidean distance* for clustering, play an important role in data mining, since they are related to the quality of the model extracted. This class of constraints specifies a requirement that the computation of a measure should respect. It involves both the knowledge about data and the knowledge about the characteristics of a model. For example, if we consider clustering people as moving objects, the trajectory implementing the shortest distance cannot cross a wall, or we can constraints a classifier to provide a minimum level of accuracy.

Starting from model constraints for classification, [YG04, VSKSvdH09] deal with the design of a classifier under constrained performance requirements. In particular, [VSKSvdH09] enables the user to define a desired classifier performance. The work provides a complete analysis when a classifier is constrained to a desired level of precision (defined as F-measure and/or to tp-/fp-rate related performance measures). The learned model is adjusted to achieve the desired performance, abstaining to classifying ambiguous examples in order to guarantee the required level of performance. Furthermore, [VSKSvdH09] studies the effect on an ROC curve when ambiguous instances are left unclassified. This is an example when a set of constraints defined on measures clearly influences also the learned model implicitly. Similarly in [YG04], an ensemble of neural networks is constrained by a given tp or fp-rate to ensure that the classification error for the most important class is within a desired limit. The final classifier is tuned by using a different structure (or architecture), employing different training samples, and training with a different subset of features for individual classifiers with respect to phase of employment. In most of the cases model constraints are used during the model construction phase.

Many papers focus on metric learning driven by constraints for clustering. Distance measure learning and clustering with constraints in K-means were both considered in [BBM04b], and the result was extended to a Hidden Markov random field formulation in [BBM04a]. In [SJ04], an SVM-like approach is employed to learn a weighted distance from relative constraints. The method learns a weighted euclidean distance from constraints by solving a convex optimization problem similar to SVMs to find the maximum margin weight vector. In this case, the approach integrates the input points with a set of training constraints that specify the distance requirements among points. Kumar and Kummamuru [KK08] proposed to learn an SVaD [KKA04] measure from relative comparisons. Relative comparisons were first employed in [SJ03] to learn distance measures using SVMs. The existing results on relative comparisons can be used to solve clustering problems with relative constraints (since each relative constraint is equivalent to two relative comparisons).

Besides those expressed on support and confidence, interestingness constraints specify thresholds on statistical measures of a pattern. We can find three kinds of interestingness measures. With *time constraints*, the user has the possibility of choosing not only the minimum support, but also time gaps and window size [SA96, PHW07, MPT09]. The former permits to constrain itemsets in a pattern to occur neither too close, nor too far w.r.t the time. Considering *recency, frequency and monetary constraints*, a model can be used to predict the behavior of a customer on the basis of history data, with the aim of analyzing how often and recently a customer purchases as well as how much he/she spends [BW95, WLW10]. Finally *aggregate constraints* are based on aggregates of items in a pattern, where the aggregate function can be sum, avg, max, min. See [ZZNS09] for a recent review on the various interestingness measures.

3.2 Hard and Soft Constraints

The use of constraints enables a mining method to explore only those solutions consistent with users expectations. Constraints may not always improve the reliability of the extracted model, e.g. data overfitting. Generally, it is not guaranteed that the use of constraints improves the reliability of the objective measures. Moreover in some cases constraints can be redundant, e.g. a constraint which does not affect the search solution space, and/or they can cause conflicts and introduce inconsistencies on final result.

For example, if we constrain two elements, say a and b, to be in the same cluster if their distance is lower than a given threshold t_1, and, at the same time, we require that a and b cannot be in the same cluster if their distance is greater than an additional threshold t_2, the satisfaction of these two constraints could not be solved by any cluster partitioning if t_2 is lower than $t_!$. Similarly, forcing a classifier to provide a desired performance can lead to find empty solutions since there is not a model extracted from the data that satisfies the required constraints, e.g. [VSKSvdH09] avoids this situation. The learned model is adjusted to achieve the desired performance by abstaining to classifying the most ambiguous example in order to guarantee the required level of performance.

Typically, these events happen when some sets of constraints work well but some others do not [Dav12]. This aspect requires the use of measures to evaluate how much a set of constraints is useful. Davidson et al. [DWB06, WBD06] introduce the concepts of *informativeness* and *coherence*. In the case of clustering, the authors define the informativeness as the amount of information in the constraint set that the algorithm cannot determine on its own. It is determined by the clustering algorithm's objective function (bias) and search preference. While given a distance matrix, the coherence measures the amount of agreement within the constraints themselves. The above definitions should be revised in the case of classification or pattern mining, but their relevance is already clear.

The above observations require that a user can define the way for computing the measure related to a constraint. Furthermore, the user expresses "how well" a constraint should be satisfied. Generally, the use of constraints does not necessarily guarantee the achievement of a solution. In order to control this effect it can be necessary to relax constraints. This leads to the need of offering the possibility of classifying constraints as either **hard** or **soft**, that is relaxable:

- **Hard constraint:** a constraint is called *hard* if a model that violates it is unacceptable. The use of only this class of constraints can involve the discovery of empty solutions. A hard-constrained algorithm halts when there does not exist a state that satisfies all the constraints, and it returns no results [OY12]. This situation is common when a large set of constraints is provided as input.
- **Soft constraint:** a constraint is called *soft* if even though a model that satisfies the constraint is preferable, a solution is acceptable anyway and especially when no any other (or better) solution is available [BMR97]. Typically, it is known that some constraints work well for finding the required solution, while others do not, and in some context where a result is needed in any case, it is important to select a set of useful constraints that should be considered as hard, while others can be treated as soft [DWB06].

This dimension is strictly related to the actual definition of a constraint and it should not be perceived as a rigid categorization. As explained above, there are some constraints that can be both hard and relaxed as soft based on the problem and the properties the solution requires.

3.3 Phase-Defined Constraints

Since a data mining task, or more generally a knowledge extraction process, is based on different iterated phases, constraints can be classified also with respect to where a knowledge extraction process can evaluate and satisfy the set of constraints defined by the user.

The *pre-processing* phase includes data cleaning, normalization, transformation, feature extraction and selection and its aim is to produce a set of data for the subsequent processing/mining step. [Pyl99] presents basic approaches for data pre-processing.

The *processing* step is the core phase where the actual knowledge extraction is performed. This is the mining phase where a model is extracted.

Finally, a *post-processing* step is required to verify if the model extracted by a data mining algorithm is valid and useful. If a model does not reach the desired standards, it is necessary to re-run the process and change parameters of the pre-processing and mining steps.

Given the above observations, techniques for constraint-driven mining can be roughly classified on the basis of the knowledge extraction phase in which they are satisfied:

- **Pre-processing constraints:** are satisfied during the pre-processing phase. They enable a restriction of the source data to the instances that can only generate patterns satisfying them.
- **Processing/Mining constraints:** are directly integrated into the mining algorithm used for extracting the model. The constraint evaluation in this case is embedded directly in the mining algorithms, enabling a reduction of the search space.
- **Post-processing constraints:** are satisfied either by filtering out patterns generated by the mining algorithm, or by highlighting only the relevant results given an interest measure provided by the user.

The phase of the knowledge extraction process where a constraint is satisfied is the last dimension we introduce. Also in this case, the above definition is useful to provide a complete picture about the use of constraints for data mining. Table 1 summarizes the main characteristics related to the different dimensions of constraints proposed in this chapter. The two main dimensions are the mining task and the kind of object where a constraint is applied. Furthermore, for each of the pairs the phase and the type of constraints are presented.

4 Conclusions: Towards New Frontiers of Data Mining

In this chapter, we presented an overview about the use of constraints in data mining. In particular, we have depicted a general multidimensional view for driving the reader into the world of constrained data mining. This chapter shows why the use of constraints is becoming an important and challenging task for the data mining community, since it requires a radical re-design of existing approaches in order to define and satisfy constraints during the whole knowledge extraction process.

Table 1. Main characteristics of the different classes of constraints

	Classification	Clustering	Pattern
Data	*phase*: pre, mining *type*: hard	*phase*: mining *type*: hard, soft	*phase*: pre, mining *type*: hard
Model	*phase*: mining, post *type*: soft	*phase*: mining *type*: soft, hard	*phase*: mining *type*: hard, soft
Measure	*phase*: mining, post *type*: hard, soft	*phase*: mining *type*: hard	*phase*: mining, post *type*: hard

Even though one of the aims of this chapter is to provide an introduction on the basic mining models and algorithms, it is worth stating that the basic concepts introduced along this overview are still valid also for advanced data mining analysis. We conclude this chapter considering the emerging phenomenon of *big data*. The final aim is to provide a set of features related to managing real data, in order to highlight that basic concepts introduced in the section of this chapter are actually the building blocks for real complex mining applications.

Often, traditional data analysis tools and techniques cannot be used because of the massive volume of data gathered by automated collection tools. The amount of data in our world has been exploding. Science gathers data at an ever-increasing rate across all scales and complexities of natural phenomena. New high-throughput scientific instruments, telescopes, satellites, accelerators, supercomputers, sensor networks and running simulations are generating massive amounts of scientific data. Companies capture trillions of bytes of information about their customers, suppliers, and operations. Smart sensing, including environment sensing, emergency sensing, people-centric sensing, smart health care, and new paradigms for communications, including email, mobile phone, social networks, blogs, Voip, are creating and communicating huge volumes of data. Sometimes, the non-traditional nature of the data implies that ordinary data analysis techniques are not applicable.

In this perspective, the challenge is particularly tough: which data mining tools are needed to master the complex dynamics of people in motion and construct concise and useful abstractions out of large volumes of mobility data is, by large, an unanswered question. Good news, hence, for researchers willing to engage in a highly interdisciplinary, highly risky and highly promising area, with a large potential impact on socially and economically relevant problems.

Big data requests a complete re-design of existing architectures and proposes new challenges on data management, privacy, and scalability among the other. Provide the *appropriate analytical* technology for distributed data mining and machine learning for big data, and a solid statistical framework adapting standard statistical data generation and analysis models to big data: once again, the sheer size and the complexity of big data call for novel analytical methods. At the same time, the kind of measures provided by the data and the population sample they describe cannot be easily modeled through standard statistical frameworks, which therefore need to be extended to capture the way the data are generated and collected.

The use of constrained-based tools, from the constraints programming to the solver, is finally under analysis from the researcher community. In this perspective, we are sure that the approaches developed along this book, generated from the experience inside the ICON project, not only represents a base for applying constrained methods to data mining but they are a first step for integrating a more versatile definition and formulation of mining approach as optimization problems by using constraint programming tools also considering the emerging phenomenon of big data.

References

[ABKS99] Ankerst, M., Breunig, M.M., Kriegel, H.-P., Sander, J.: Optics: ordering points to identify the clustering structure. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, SIGMOD 1999, pp. 49–60. ACM, New York, NY, USA (1999)

[ABR64] Aizerman, M.A., Braverman, E.A., Rozonoer, L.: Theoretical foundations of the potential function method in pattern recognition learning. Autom. Remote Control **25**, 821–837 (1964)

[AFGY02] Ayres, J., Flannick, J., Gehrke, J., Yiu, T.: Sequential pattern mining using a bitmap representation. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 429–435 (2002)

[AS94] Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of 20th International Conference on Very Large Data Bases (VLDB 1994), Santiago de Chile, Chile, 12–15 September, pp. 487–499 (1994)

[ATJ+12] Ahmed, C.F., Tanbeer, S.K., Jeong, B.-S., Lee, Y.-K., Choi, H.-J.: Single-pass incremental and interactive mining for weighted frequent patterns. Expert Syst. Appl. **39**(9), 7976–7994 (2012)

[BBD00] Bradley, P.S., Bennett, K.P., Demiriz, A.: Constrained k-means clustering. Technical report, MSR-TR-2000-65, Microsoft Research (2000)

[BBM04a] Basu, S., Bilenko, M., Mooney, R.J.: A probabilistic framework for semi-supervised clustering. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 59–68 (2004)

[BBM04b] Bilenko, M., Basu, S., Mooney, R.J.: Integrating constraints and metric learning in semi-supervised clustering. In: Proceedings of the Twenty-First International Conference on Machine Learning, ICML 2004, p. 11. ACM, New York (2004)

[BCCG12] Baralis, E., Cagliero, L., Cerquitelli, T., Garza, P.: Generalized association rule mining with constraints. Inf. Sci. **194**, 68–84 (2012)

[BDW08] Basu, S., Davidson, I., Wagstaff, K.L.: Constrained Clustering: Advances in Algorithms, Theory, and Applications. Chapman and Hall/CRC, Boca Raton (2008)

[Ber91] Bertsekas, D.P.: Linear Network Optimization - Algorithms and Codes. MIT Press, Cambridge (1991)

[BFOS84] Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth International Group, Belmont (1984)

[BG06] Banerjee, A., Ghosh, J.: Scalable clustering algorithms with balancing constraints. Data Min. Knowl. Discov. **13**(3), 365–395 (2006)

[BG08] Banerjee, A., Ghosh, J.: Clustering with balancing constraints. Constrained Clustering: Advances in Algorithms. Theory, and Applications, pp. 171–200. Chapman and Hall/CRC, Boca Raton (2008)

[BGV92] Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT 1992, pp. 144–152. ACM, New York (1992)

[BK03] Barbará, D., Kamath, C. (eds.): Proceedings of the Third SIAM International Conference on Data Mining, 1–3 May 2003. SIAM, San Francisco (2003)

[BMR97] Bistarelli, S., Montanari, U., Rossi, F.: Semiring-based constraint solving and optimization. J. ACM **44**(2), 201–236 (1997)

[Bur98] Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Min. Knowl. Discov. **2**(2), 121–167 (1998)

[BW95] Bult, J.R., Wansbeek, T.J.: Optimal selection for direct mail. Mark. Sci. **14**(4), 378–394 (1995)

[CMB03] Capelle, M., Masson, C., Boulicaut, J.F.: Mining frequent sequential patterns under regular expressions: a highly adaptive strategy for pushing constraints. In: Proceedings of the Third SIAM International Conference on Data Mining, pp. 316–320 (2003)

[CTG12] Chand, C., Thakkar, A., Ganatra, A.: Sequential pattern mining: survey and current research challenges. Int. J. Soft Comput. Eng. (IJSCE) **2**(1), 2231–2307 (2012)

[CV95] Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)

[Dav12] Davidson, I.: Two approaches to understanding when constraints help clustering. In: The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 1312–1320 (2012)

[DBB08] Demiriz, A., Bennett, K.P., Bradley, P.S.: Using assignment constraints to avoid empty clusters in k-means clustering. Constrained Clustering: Advances in Algorithms. Theory, and Applications, pp. 201–220. Chapman and Hall/CRC, Boca Raton (2008)

[DDV13] Dao, T.-B.-H., Duong, K.-C., Vrain, C.: A declarative framework for constrained clustering. In: Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (eds.) ECML PKDD 2013. LNCS (LNAI), vol. 8190, pp. 419–434. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40994-3_27

[DH73] Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. Wiley, New York (1973)

[DMM08] Druck, G., Mann, G.S., McCallum, A.: Learning from labeled features using generalized expectation criteria. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pp. 595–602 (2008)

[DP96] Domingos, P., Pazzani, M.J.: Beyond independence: conditions for the optimality of the simple Bayesian classifier. In: Proceedings of the 13th International Conference on Machine Learning (ICML 1996), Bari, Italy, pp. 148–156 (1996)

[DR05] Davidson, I., Ravi, S.S.: Clustering with constraints: feasibility issues and the k-means algorithm. In: Proceedings of the SIAM International Conference on Data Mining (SDM) (2005)

[DR06] Davidson, I., Ravi, S.S.: Identifying and generating easy sets of constraints for clustering. In: Proceedings of the Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference (AAAI), pp. 336–341 (2006)

[DR07] Davidson, I., Ravi, S.S.: The complexity of non-hierarchical clustering with instance and cluster level constraints. Data Min. Knowl. Discov. **14**(1), 25–61 (2007)

[DR09] Davidson, I., Ravi, S.S.: Using instance-level constraints in agglomerative hierarchical clustering: theoretical and empirical results. Data Min. Knowl. Discov. **18**(2), 257–282 (2009)

[DWB06] Davidson, I., Wagstaff, K.L., Basu, S.: Measuring constraint-set utility for partitional clustering algorithms. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 115–126. Springer, Heidelberg (2006). doi:10.1007/11871637_15

[EKSX96] Ester, M., Kriegel, H.-P., Sander, J., Xiaowei, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD), pp. 226–231 (1996)

[EZ13] Elloumi, M., Zomaya, A.Y.: Biological Knowledge Discovery Handbook: Preprocessing, Mining and Postprocessing of Biological Data, 1st edn. Wiley, New York (2013)

[FH95] Yongjian, F., Han, J.: Meta-rule-guided mining of association rules in relational databases. In: Proceedings of the Post-Conference Workshops on Integration of Knowledge Discovery in Databases with Deductive and Object-Oriented Databases (KDOOD/TDOOD), pp. 39–46 (1995)

[GMN+15] Grossi, V., Monreale, A., Nanni, M., Pedreschi, D., Turini, F.: Clustering formulation using constraint optimization. In: Bianculli, D., Calinescu, R., Rumpe, B. (eds.) SEFM 2015. LNCS, vol. 9509, pp. 93–107. Springer, Heidelberg (2015). doi:10.1007/978-3-662-49224-6_9

[GRS99] Garofalakis, M.N., Rastogi, R., Shim, K.: SPIRIT: Sequential pattern mining with regular expression constraints. In: Proceedings of 25th International Conference on Very Large Data Bases (VLDB), pp. 223–234 (1999)

[GS11] Grossi, V., Sperduti, A.: Kernel-based selective ensemble learning for streams of trees. In: Walsh, T. (ed.) IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, 16–22 July 2011, pp. 1281–1287. IJCAI/AAAI (2011)

[GT12] Grossi, V., Turini, F.: Stream mining: a novel architecture for ensemble-based classification. Knowl. Inf. Syst. **30**(2), 247–281 (2012)

[HCXY07] Han, J., Cheng, H., Xin, D., Yan, X.: Frequent pattern mining: current status and future directions. Data Min. Knowl. Discov. **15**(1), 55–86 (2007)

[HF99] Han, J., Fu, Y.: Mining multiple-level association rules in large databases. IEEE Trans. Knowl. Data Eng. **11**(5), 798–805 (1999)

[HK12] Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2012)

[HLN99] Han, J., Lakshmanan, L.V.S., Ng, R.T.: Constraint-based multidimensional data mining. IEEE Comput. **32**(8), 46–50 (1999)

[HPRZ02] Har-Peled, S., Roth, D., Zimak, D.: Constraint classification: a new approach to multiclass classification. In: Proceedings of the 13th International Conference Algorithmic Learning Theory (ALT), pp. 365–379 (2002)

[HPY00] Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, Texas, USA, 16–18 May, pp. 1–12 (2000)

[IWM00] Inokuchi, A., Washio, T., Motoda, H.: An apriori-based algorithm for mining frequent substructures from graph data. In: Zighed, D.A., Komorowski, J., Żytkow, J. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 13–23. Springer, Heidelberg (2000). doi:10.1007/3-540-45372-5_2

[Jen96] Jensen, F.V.: An introduction to Bayesian networks. Springer, New York (1996)

[KK08] Kumar, N., Kummamuru, K.: Semisupervised clustering with metric learning using relative comparisons. IEEE Trans. Knowl. Data Eng. **20**(4), 496–503 (2008)

[KKA04] Kummamuru, K., Krishnapuram, R., Agrawal, R.: Learning spatially variant dissimilarity (SVaD) measures. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 611–616 (2004)

[LLS00] Lin, T.S., Loh, W.Y., Shib, Y.S.: A comparison of prediction accuracy, complexity, and training time of thirty-tree old and new classification algorithms. Mach. Learn. **40**(3), 203–228 (2000)

[LYC08] Li, Y.-C., Yeh, J.-S., Chang, C.-C.: Isolated items discarding strategy for discovering high utility itemsets. Data Knowl. Eng. **64**(1), 198–217 (2008)

[MAR96] Mehta, M., Agrawal, R., Rissanen, J.: SLIQ: A fast scalable classifier for data mining. In: Proceedings of 5th International Conference on Extending Database Technology (EBDT 1996), Avignon, France, pp. 18–32 (1996)

[ME10] Mabroukeh, N.R., Ezeife, C.I.: A taxonomy of sequential pattern mining algorithms. ACM Comput. Surv. **43**(1), 3: 1–3: 41 (2010)

[Mic97] Michell, T.: Machine Learning. McGraw Hill, New York (1997)

[Mor82] Moret, B.M.E.: Decision trees and diagrams. Comput. Surv. **14**(4), 593–623 (1982)

[MPT09] Masseglia, F., Poncelet, P., Teisseire, M.: Efficient mining of sequential patterns with time constraints: reducing the combinations. Expert Syst. Appl. **36**(2), 2677–2690 (2009)

[NF07] Nijssen, S., Fromont, É.: Mining optimal decision trees from itemset lattices. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 530–539 (2007)

[NF10] Nijssen, S., Fromont, E.: Optimal constraint-based decision tree induction from itemset lattices. Data Min. Knowl. Discov. Fromont. **21**(1), 9–51 (2010)

[NPS00] Niyogi, P., Pierrot, J.-B., Siohan, O.: Multiple classifiers by constrained minimization. In: Proceedings of the Acoustics, Speech, and Signal Processing of 2000 IEEE International Conference on ICASSP 2000, vol. 06, pp. 3462–3465. IEEE Computer Society, Washington, DC (2000)

[OY12] Okabe, M., Yamada, S.: Clustering by learning constraints priorities. In: Proceedings of the 12th International Conference on Data Mining (ICDM2012), pp. 1050–1055 (2012)

[PF08] Park, S.H., Furnkranz, J.: Multi-label classification with label constraints. Technical report, Knowledge Engineering Group, TU Darmstadt (2008)

[PHMA+04] Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U., Hsu, M.: Mining sequential patterns by pattern-growth: the prefixspan approach. IEEE Trans. Knowl. Data Eng. **16**(11), 1424–1440 (2004)

[PHW07] Pei, J., Han, J., Wang, W.: Constraint-based sequential pattern mining: the pattern growth methods. J. Intell. Inf. Syst. **28**(2), 133–160 (2007)

[Pyl99] Pyle, D.: Data Preparation for Data Mining. Morgan Kaufmann Publishers Inc., San Francisco (1999)

[Qui86] Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**, 81–106 (1986)

[Qui93] Quinlan, J.R.: C4.5 Programs for Machine Learning. Wadsworth International Group, Belmont (1993)

[Qui96] Quinlan, J.R.: Improved use of continuous attributes in C4.5. J. Artif. Intell. Res. **4**, 77–90 (1996)

[Rug02] Ruggieri, S.: Efficient C4.5. IEEE Trans. Knowl. Data Eng. **14**(2), 438–444 (2002)

[SA95] Srikant, R., Agrawal, R.: Mining generalized association rules. In: Proceedings of the 21st Conference on Very Large Data Bases (VLDB), pp. 407–419 (1995)

[SA96] Srikant, R., Agrawal, R.: Mining sequential patterns: generalizations and performance improvements. In: Proceedings of the 5th International Conference on Extending Database Technology (EDBT), pp. 3–17 (1996)

[SAM96] Shafer, J., Agrawal, R., Mehta, M.: Sprint: a scalable parallel classifier for data mining. In: Proceedings of 1996 International Conference on Very Large Data Bases (VLDB 1996), Bombay, India, pp. 544–555 (1996)

[SC08] Steinwart, I., Christmann, A.: Support Vector Machines, 1st edn. Springer Publishing Company, Incorporated, Heidelberg (2008)

[SG03] Strehl, A., Ghosh, J.: Relationship-based clustering and visualization for high-dimensional data mining. INFORMS J. Comput. **15**(2), 208–230 (2003)

[Sha09] Shankar, S.: Utility sentient frequent itemset mining and association rule mining: a literature survey and comparative study. Int. J. Soft Comput. Appl. **4**, 81–95 (2009)

[SJ03] Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: Proceedings of Conference Advances in Neural Information Processing Systems (NIPS) (2003)

[SJ04] Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: NIPS, MIT Press (2004)

[SON95] Savasere, A., Omiecinski, E., Navathe, S.B.: An efficient algorithm for mining association rules in large databases. In: Proceedings of the 21st International Conference on Very Large Data Bases (VLDB), Zurich, Switzerland, 11–15 September 1995, pp. 432–444 (1995)

[ST02] Sriphaew, K., Theeramunkong, T.: A new method for finding generalized frequent itemsets in generalized association rule mining. In: Proceedings of the 7th IEEE Symposium on Computers and Communications (ISCC), pp. 1040–1045 (2002)

[SVA97] Srikant, R., Quoc, V., Agrawal, R.: Mining association rules with item constraints. In: Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD), pp. 67–73 (1997)

[TJHA05] Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. J. Mach. Learn. Res. **6**, 1453–1484 (2005)

[TM03] Tao, F., Murtagh, F.: Weighted association rule mining using weighted support and significance framework. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 661–666 (2003)

[Toi96] Toivonen, H.: Sampling large databases for association rules. In: Proceedings of the 22nd International Conference on Very Large Data Bases (VLDB), Mumbai (Bombay), India, 3–6 September, pp. 134–145 (1996)

[TSK06] Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison Wesley, Boston (2006)

[TSWYng] Tseng, V.S., Shie, B.-E., Wu, C.-W., Philip, S.: Efficient algorithms for mining high utility itemsets from transactional databases. IEEE Transactions on Knowledge and Data Engineering, forthcoming

[VSKSvdH09] Vanderlooy, S., Sprinkhuizen-Kuyper, I.G., Smirnov, E.N., Jaap van den Herik, H.: The ROC isometrics approach to construct reliable classifiers. Intell. Data Anal. **13**(1), 3–37 (2009)

[WBD06] Wagstaff, K., Basu, S., Davidson, I.: When is constrained clustering beneficial, and why? In: Proceedings of The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference (AAAI) (2006)

[WC00] Wagstaff, K., Cardie, C.: Clustering with instance-level constraints. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on on Innovative Applications of Artificial Intelligence (AAAI/IAAI), p. 1097 (2000)

[WCRS01] Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained k-means clustering with background knowledge. In: Proceedings of the Eighteenth International Conference on Machine Learning, ICML 2001, pp. 577–584. Morgan Kaufmann Publishers Inc., San Francisco (2001)

[WFH11] Witten, I.H., Frank, E., Hall, M.: Data Mining, Pratical Machine Learning Tools and Techiniques, 3rd edn. Morgan Kaufmann, San Francisco (2011)

[WH11] Wu, C.-M., Huang, Y.-F.: Generalized association rule mining using an efficient data structure. Expert Syst. Appl. **38**(6), 7277–7290 (2011)

[WK91] Weiss, S.M., Kulikowski, C.A.: Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets. Machine Learning and Expert Systems. Morgan Kaufmann, San Francisco (1991)

[WLW10] Wei, J.-T., Lin, S.-Y., Hsin-Hung, W.: A review of the application of RFM model. Afr. J. Bus. Manag. **4**(19), 4199–4206 (2010)

[WWZ+05] Wang, W., Wang, C., Zhu, Y., Shi, B., Pei, J., Yan, X., Han, J.: Graphminer: a structural pattern-mining system for large disk-based graph databases and its applications. In: zcan, F. (ed.) SIGMOD Conference, pp. 879–881. ACM (2005)

[WYY00] Wang, W., Yang, J., Philip, S.: Efficient mining of weighted association rules (WAR). In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 270–274 (2000)

[YG04] Yan, W., Goebel, K.F.: Designing classifier ensembles with constrained performance requirements. In: Proceedings of SPIE Defense and Security Symposium, Multisensor Multisource Information Fusion: Architectures, Algorithms, and Applications 2004, pp. 78–87 (2004)

[YH02] Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: Proceedings of the 2002 IEEE International Conference on Data Mining, ICDM 2002, p. 721. IEEE Computer Society, Washington, DC, USA (2002)

[YL05] Yun, U., Leggett, J.J.: WFIM: weighted frequent itemset mining with a weight range and a minimum weight. In: Proceeding of the 2005 SIAM International Data Mining Conference, Newport Beach, CA, pp. 636–640 (2005)

[YR11] Yun, U., HoRyu, K.: Approximate weighted frequent pattern mining with/without noisy environments. Knowl.-Based Syst. **24**(1), 73–82 (2011)

[YSRY12] Yun, U., Shin, H., Ho Ryu, K., Yoon, E.: An efficient mining algorithm for maximal weighted frequent patterns in transactional databases. Knowl.-Based Syst. **33**, 53–64 (2012)

[Zak01] Zaki, M.J.: SPADE: an efficient algorithm for mining frequent sequences. Mach. Learn. **42**(1/2), 31–60 (2001)

[ZG03] Zhong, S., Ghosh, J.: Scalable, balanced model-based clustering. In: Proceedings of the Third SIAM International Conference on Data Mining, San Francisco (SDM) (2003)

[ZPOL97] Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New algorithms for fast discovery of association rules. In: Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD 1997), Newport Beach, California, USA, 14–17 August, pp. 283–286 (1997)

[ZRL96] Zhang, T., Ramakrishnan, R., Livny, M.: Birch: an efficient data clustering method for very large databases. SIGMOD Rec. **25**(2), 103–114 (1996)

[ZZ02] Zhang, C., Zhang, S.: Association Rule Mining: Models and Algorithms. LNCS, vol. 2307. Springer, Heidelberg (2002)

[ZZNS09] Zhang, Y., Zhang, L., Nie, G., Shi, Y.: A survey of interestingness measures for association rules. In: Proceedings of the Second International Conference on Business Intelligence and Financial Engineering (BIFE), pp. 460–463 (2009)

http://www.springer.com/978-3-319-50136-9

Data Mining and Constraint Programming
Foundations of a Cross-Disciplinary Approach
Bessiere, C.; De Raedt, L.; Kotthoff, L.; Nijssen, S.;
O'Sullivan, B.; Pedreschi, D. (Eds.)
2016, XII, 349 p. 73 illus., Softcover
ISBN: 978-3-319-50136-9