

Master's Thesis

A Scientific Framework Application for Testing Knowledge Discovery Methods

Mario Zupan

Institute for Information Systems and Computer Media,
Graz University of Technology



Supervisor: Assoc. Prof. Dr. Andreas HOLZINGER, PhD, MSc, MPh, BEng, CEng,
DipEd, MBCS

Graz, October 9, 2012

Masterarbeit

(Diese Arbeit ist in englischer Sprache verfasst)

Ein universelles wissenschaftliches Computerprogramm zum Testen von Wissenserschließungsmethoden

Mario Zupan

Institut für Informationssysteme und Computer Medien,
Technische Universität Graz



Betreuer: Univ.-Doz. Dr. Andreas HOLZINGER, PhD, MSc, MPh, BEng, CEng, DipEd,
MBCS

Graz, 9. Oktober 2012

Abstract

The field of knowledge discovery and its applications are vast and widely studied, however, for researchers who are unexperienced in the use of even relatively popular knowledge discovery methods, it can be difficult to find useful, practically oriented literature, which does not require extensive previous knowledge of the general subject. It can be even harder for inexperienced users, to actually use these methods to generate useful information. This thesis will provide a general overview and description of state-of-the-art knowledge discovery methods with a focus on their advantages, disadvantages, applications and existing software in the real world. The thesis also proposes 'KNODWAT', a web-based, highly extensible scientific framework application for testing knowledge discovery methods, making it possible for researchers who are new to the field of knowledge discovery and data mining in general, to test existing methods on their data, to find out whether and to what extent these are suitable and how well they perform.

Keywords

Knowledge Discovery, Data Mining, Machine learning, Human-Computer Interaction, Data Visualization, Data Analytics

ÖSTAT classification

1108, 1105, 1138, 1122

ACM classification

H.2.8, H.3.5, I.2.6

Kurzfassung

Das Gebiet der Wissenserschließung und dessen Anwendungsbereiche sind gewaltig und weitreichend studiert, aber für Forscher mit wenig Erfahrung in der Anwendung sogar relativ bekannter Wissenserschließungsmethoden, ist es oft schwierig nützliche, praktisch orientierte Literatur zu finden, die kein ausgeprägtes Vorwissen auf dem generellen Gebiet erfordert. Es kann sogar noch schwieriger sein für neue Benutzer, diese Methoden effizient zu benutzen um brauchbare Daten zu erzeugen. Diese Diplomarbeit stellt einen generellen Überblick und eine Beschreibung der in der Literatur gängigsten Wissenserschließungsmethoden dar, fokussiert auf deren Vorteile, Nachteile, Anwendungsbereiche und auf bereits bestehende Software auf dem Gebiet. Die Diplomarbeit stellt außerdem 'KNODWAT' vor, ein webbasiertes, stark erweiterbares Computerprogramm für das Testen von Wissenserschließungsmethoden, die es Forschern mit wenig Erfahrung im Bereich der theoretischen Wissenserschließungsmethoden erleichtern soll, vorhandene Methoden anhand ihrer Daten zu testen, deren Leistungsfähigkeit in diesem Kontext zu untersuchen und sowohl die Bewertung als auch die Auswahl dieser zu ermöglichen.

Schlüsselwörter

Wissenserschließung, Data Mining, Maschinelles Lernen, Mensch-Computer Interaktion, Datenvisualisierung, Datenanalyse

ÖSTAT classification

1108, 1105, 1138, 1122

ACM classification

H.2.8, H.3.5, I.2.6

This page intentionally left blank

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, DATE

Mario Zupan

Acknowledgements

First and foremost, I want to thank my parents Gertrud and Martin for their unyielding support, patience and understanding during the course of my studies at the TUGraz. I also want to thank my sister Daniela, for her counsel and for having been both an inspiration and a role model for me in the past. Last but not least, I want to express my deep gratitude to Prof. Andreas Holzinger, my supervisor, who was an absolute pleasure to work with and whose amazing passion, expertise and guidance were a fundamental factor in the success of this thesis.

Thank you.

Mario Zupan
Graz, October 9, 2012

This page intentionally left blank

Table of Contents

1	Introduction and Motivation for Research	15
1.1	Objectives and Motivation	15
1.2	Organization of the Thesis	17
2	Theoretical Background	19
2.1	Supervised Methods	19
2.1.1	Classification Trees	22
2.1.2	Support Vector Machines	30
2.1.3	Bayesian Networks	36
2.1.4	Regression Analysis Methods	42
2.2	Unsupervised Methods	47
2.2.1	Clustering Algorithms	48
2.2.2	Link Analysis	55
2.2.3	Association Rules	62
2.2.4	Constraint-based Methods	67
2.3	Others	71
2.3.1	Evolutionary Methods	72
2.3.2	Neural Networks	80
2.3.3	Swarm Intelligence Methods	87
3	Related Work	91
3.1	Literature	91

3.1.1	Books	92
3.1.2	Journal Papers	93
3.2	Software	95
4	Materials and Methods	101
4.1	Technology	102
4.1.1	Spring	102
4.1.2	Hibernate	103
4.1.3	jQuery	103
4.1.4	Bootstrap	104
4.1.5	Others	104
4.1.6	Development Tools	105
4.2	Application Structure	106
4.2.1	Data model	107
4.2.2	Program structure	111
4.2.3	Frontend	116
4.2.4	Functionality	120
4.2.5	Technical Details	132
4.2.6	Testing	141
4.2.7	Extensibility	142
5	Results	147
5.1	KNODWAT setup	148
5.1.1	Configuration	148
5.1.2	Building with Apache Maven	148
5.1.3	Deployment on Apache Tomcat	149
5.2	Benchmark	150
5.2.1	Test data	150
5.2.2	Test results	152
5.2.3	Analysis	160

6	Conclusions	167
7	Future Work	169
	List of Figures	173
	List of Tables	179
	References	181

This page intentionally left blank

1. Introduction and Motivation for Research

1.1 Objectives and Motivation

With the increasing availability of large data sets in every different area of science as well as the computational power to process them, computational knowledge discovery methods become more and more important. There are, however, many different methods with strengths in some and weaknesses in other areas. These methods are used to find patterns, relationships and other relevant information inside of highly complex data sets, which can greatly increase the efficiency of research in many different areas of science beyond just information technologies such as medicine and the natural sciences in general. The process of knowledge discovery and its implications have been studied for some time. One of the most popular representations of the complete process, especially showing how knowledge discovery does not only consist of mining data, is the one of Fayyad et al. (1996) as shown in figure 1.1.

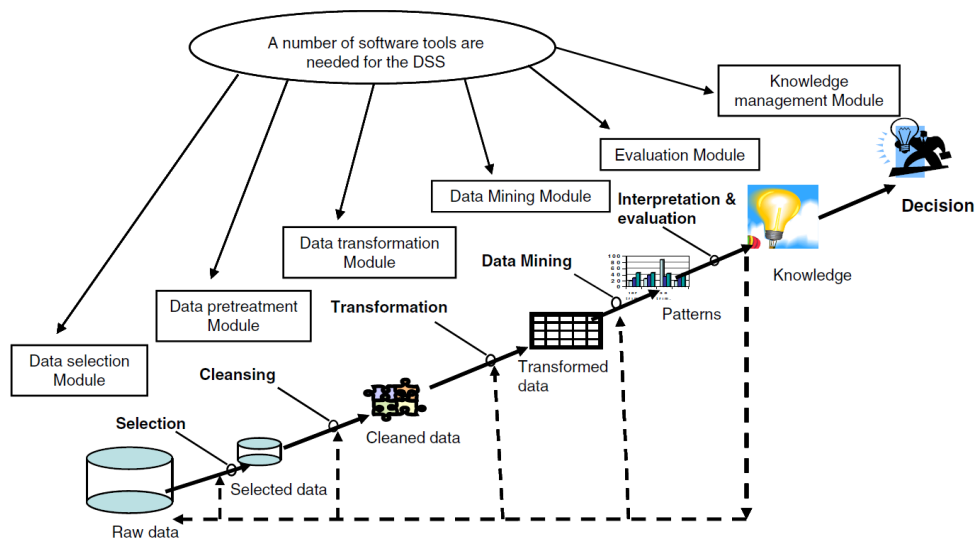


Figure 1.1: The Knowledge Discovery Process (Fayyad et al., 1996)

This thesis aims at providing an overview of the most popular computational knowledge discovery methods, with a focus on making the different algorithms comparable by describing their strengths, weaknesses and optimal areas of application as well as existing software. Researchers may have difficulties in finding easily understandable and practically usable information about the multiple available methods for knowledge discovery if they don't possess an extensive knowledge in machine learning. This situation needs to be improved, as every area of science can greatly benefit from using the thoroughly studied and implemented algorithms to get new insights into parts of the endless amounts of data available nowadays. Furthermore, this thesis proposes 'KNODWAT', a highly extensible scientific framework application for testing different knowledge discovery methods. The application provides features to administrate and manage projects as well as users and social features including sharing and commenting on data. Moreover, it can be easily extended in various areas, not only, but most importantly by adding new knowledge discovery methods, thus enabling researchers to test their data with diverse methods in a very easy and intuitive way, without having to know the insides of the actual algorithms behind these methods and also to compare the results in order to choose the most suitable method for the data-set at hand. This thesis should serve as a good example of combining both science and engineering (Holzinger, 2010), considering the approach that "Science is to test ideas, engineering is to put these ideas into practice" (<http://www.hci4all.at/>).

1.2 Organization of the Thesis

Chapter 2 provides a theoretical background of the topic of knowledge discovery methods in the form of a survey. It provides a general overview of the most popular knowledge discovery methods and is divided into supervised and unsupervised methods. These methods and their basic functionalities are described with a focus on application areas, advantages and disadvantages for each of them. The survey is targeted at researchers with limited experience in machine learning, aiming at making prominent knowledge discovery methods more available to people from different scientific disciplines including medicine, biology and the life sciences.

Chapter 3 Related Work provides an overview of relevant literature in the field as well as an introduction to existing software in the area of knowledge discovery with a focus on open source frameworks.

Chapter 4 Materials and Methods describes the software-engineering methods used to create the 'KNODWAT'-framework in a detailed and thorough way, providing the reader with a description of used applied frameworks, relevant structural diagrams and technical details for the more complex parts of the software. The reasoning behind architectural and technical decisions is available in this chapter as well.

Chapter 5 Results addresses the KNODWAT application, which has been created during the course of this thesis, presenting the software in a productive environment and some setup guidelines. The chapter also includes a basic benchmark of the two implemented algorithms (CART and C4.5) within the 'KNODWAT'-framework, comparing their generated results on a simple dataset and in the course of that, showing how the features of 'KNODWAT' can provide helpful assistance for researchers.

Chapter 6 Conclusions draws some concluding remarks regarding the created software and the problems it aimed to help solve.

Chapter 7 Future Work provides a list of features, fixes and extensions for the 'KNODWAT'-framework as well as a general outlook on how to further improve the accessibility of knowledge discovery methods for researchers of different scientific disciplines.

2. Theoretical Background

2.1 Supervised Methods

Supervised Learning Methods or Predictive Methods are a fundamental concept in the field of machine learning and data mining. Supervised methods aim at building behavioral models in order to discover relationships between independent input attributes and dependent target attributes. There are two different kinds of methods, the ones that are built to predict quantifiable real-value classes for the input space, called Regression Models and the methods focused on mapping the input space into predetermined, non quantifiable classes, called Classification Models. There are a number of representations for supervised classification and regression. The following part of this thesis aims at providing a short overview and survey of some of these methods with references to more detailed literature on the subject.

2.1.0.1 The Process of Supervised Learning

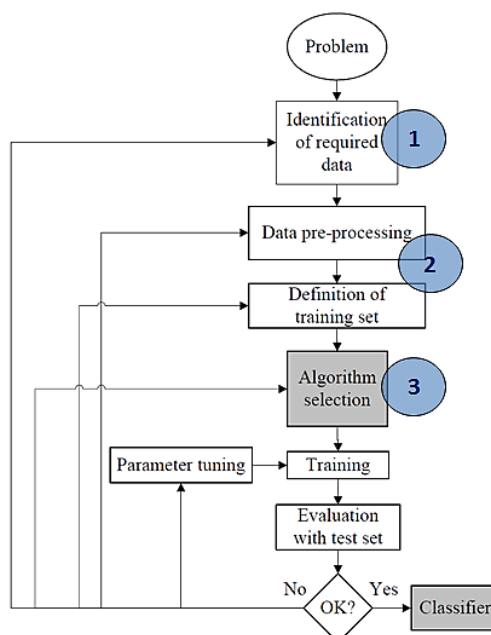


Figure 2.1: The Process of Machine Learning (Kotsiantis, 2007)
Taken from the slides of Holzinger (2012a)

Figure 2.1 shows the process of supervised machine learning with regard to a real-world problem, as described in (Kotsiantis, 2007). The process consists of multiple steps, with (1) Identification of required data, (2) Data pre-processing and (3) Algorithm selection are the most relevant ones with respect to giving an overview of the general process of supervised learning.

- **Identification of required data:**

In this step, the problem has already been well defined, but in order to perform a classification using a supervised learning method, a dataset is required. In some cases, there are experts who can fill this gap by providing data they have collected on the subject. Another way to accumulate data is to measure in a "brute-force" kind of way, which means that every piece of information that could somehow be connected to the target attributes, gets collected. Obviously, this method of data collecting can lead to very faulty, noisy data with missing values, which makes data pre-processing an important step (Zhang et al., 2003).

- **Data pre-processing:**

If a suitable data set has been found, but some values are missing, or the data is noisy and / or faulty in general, there are many methods to improve the quality of the data set and, as a consequence, the classifier. A number of different approaches and methods are available to handle missing data (Batista and Carolina Monard, 2003) and noise, including Instance selection (Liu and Motoda, 2001). Sampling of large datasets (Reinartz, 2002) as well as feature subset selection, a method applied to remove irrelevant features, (Yu and Liu, 2004) are also very effective tools to increase the quality of a dataset and, in some cases, even reduce its dimensionality and complexity.

- **Definition of training set:**

In typical supervised learning methods, the data set is split up into two parts, namely a training set and a test set. The basic idea is, to create the classifier using the training set and testing that classifier on the test set. The selection of, which data goes into which set usually happens randomly, in order not to create a bias in the classifier by picking instances manually.

- **Algorithm selection:**

The selection of the algorithm to perform supervised learning can be considered the most important and critical step during the process. A classifier's performance is usually evaluated on the basis of the generalization error, its computational complexity and its comprehensibility. Computational complexity can be measured fairly easily, by calculating or testing the CPU-consumption for the inducer in use. Sometimes, in cases where the data set is not of substantial size, this criterion can be ignored. Comprehensibility on the other hand is quite hard to measure objectively, but it can be useful for researchers to visualize and understand how the classifier works, for example in the case of deducting rules from a classification tree 2.1.1. The generalization error can be estimated by using the training error, but also empirically, by methods such as sub-sampling and n-fold cross validation, enabling the re-sampling of the training set and the calculation of an average generalization error for all the different partitions. More information about this subject can be found in (Maimon and Rokach, 2010).

2.1.1 Classification Trees

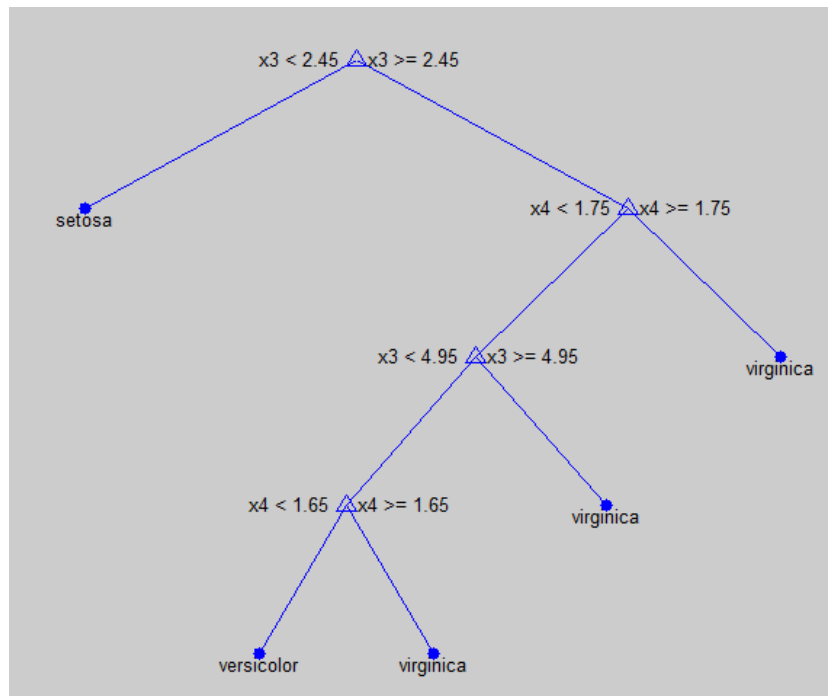


Figure 2.2: A simple Decision Tree created in Matlab using CART

Decision trees have been used as a decision support system for a long period of time and in many different fields. In the context of classification, decision trees in general, are rooted, directed trees with two different kinds of nodes. There are internal nodes, which split the instance space based on feature attributes and there are leaves, which represent classifications. To perform classification using a decision tree, an instance starts at the root and is questioned at every internal node until it reaches a leaf, implying that the path that was taken by an instance until it reached a classification can be formulated as a rule, thus making decision trees very comprehensible for humans. There are multiple tree inducers, or algorithms, that use different criteria and optimization methods in order to find a minimal decision tree with minimized generalization error. Usually this happens in a two-step process, where in the first step, the decision tree is grown in a recursive, top-down and greedy manner until a stopping criterion is met. This tree is then pruned in order to simplify it without increasing the generalization error as well as to improve its performance and comprehensibility. Decision tree classification is a useful tool for many domains such as medicine, manufacturing and data mining in general.

2.1.1.1 Splitting and Stopping Criteria

Decision trees use univariate and multivariate splitting criteria in order to find the optimal nodes for splitting, with univariate criteria being most commonly used. Finding good multivariate criteria is much more difficult and is usually achieved by a linear combination of input attributes. For univariate splitting criteria there are many approaches, the most popular ones in literature are described here:

Information Gain

The Information Gain Criterion uses the entropy measure ($Entropy(y, S)$) to determine the impurity of the regarded attribute. As described in (Quinlan, 1987), it is defined as (Maimon and Rokach, 2010):

$$InformationGain(a_i, S) = Entropy(y, S) - \sum_{v_{i,j} \in dom(a_i)} \frac{|\sigma_{a_i = v_{i,j}} S|}{|S|} Entropy(y, \sigma_{a_i = v_{i,j}} S) \quad (2.1)$$

Gini Index

The Gini Index Criterion is impurity-based as well, but it uses the Gini index (or Gini ratio), which is a statistical measure developed by Corrado Gini in 1912. It is a measure for inequalities between elements of a distribution. The Gini coefficient and the criterion for choosing an attribute are defined as (Maimon and Rokach, 2010):

$$Gini(y, S) = 1 - \sum_{c_j \in dom(y)} \left(\frac{|\sigma_y = c_j S|}{|S|} \right)^2 \quad (2.2)$$

$$GiniGain(a_i, S) = Gini(y, S) - \sum_{v_{i,j} \in dom(a_i)} \frac{|\sigma_{a_i = v_{i,j}} S|}{|S|} Gini(y, \sigma_{a_i = v_{i,j}} S) \quad (2.3)$$

Gain Ratio

The Gain Ratio Criterion evolved from the Information Gain Criterion by normalizing the information gain. It is described in (Quinlan, 1993) and is defined as (Maimon and Rokach, 2010):

$$GainRatio(a_i, S) = \frac{InformationGain(a_i, S)}{Entropy(a_i, S)} \quad (2.4)$$

Twoing Criterion

The Twoing Criterion is a binary criterion which performs well for widely spread target attributes (Breiman et al., 1984). It is defined as (Maimon and Rokach, 2010):

$$twoing(a_i, dom_1(a_i), dom_2(a_i), S) = 0.25 \frac{|\sigma_{a_i} \in dom_1(a_i)S|}{|S|} \frac{|\sigma_{a_i} \in dom_2(a_i)S|}{|S|} \left(\sum_{c_i \in dom(y)} \left| \frac{|\sigma_{a_i} \in dom_1(a_i)ANDy = c_iS|}{|\sigma_{a_i} \in dom_1(a_i)S|} - \frac{|\sigma_{a_i} \in dom_2(a_i)ANDy = c_iS|}{|\sigma_{a_i} \in dom_2(a_i)S|} \right| \right)^2 \quad (2.5)$$

Others

There are of course other univariate criteria such as AUC (Area Under Curve) (Ferri et al., 2002), ORT (Orthogonal Criterion) (Fayyad and Irani, 1992) and DKM (Dietterich, Kearns, Mansour) (Kearns and Mansour, 1996) and (Dietterich et al., 1996), but the criteria mentioned above are the ones used by the most popular tree inducers found in literature. As for Multivariate Splitting Criteria, they use a linear combination to join input attributes. Different ways to achieve this are described in (Breiman et al., 1984), (Duda and Hart, 1973a) and (Bennett et al., 1992) among others.

Performance

There have been studies comparing these criteria, among them (Ben-Bassat, 1978), (Buntine and Niblett, 1992), (Baker and Jain, 1976) and (Lim et al., 2000), but there was no consensus on which of the criteria can be considered the overall "best" criterion. Rather one came to the conclusion, that the chosen splitting criteria would not affect the tree performance greatly most of the time due to the fact of no criterion being dominant in all or most cases.

Stopping Criteria

Another set of criteria used by decision trees are stopping criteria, which determine under which conditions the algorithm stops growing, thus dictating the complexity and size of the decision tree. These criteria vary greatly between cases, but usually consist of an obtained final-state or certain thresholds based on splitting criteria or the depth of the tree.

2.1.1.2 Tree Pruning

After the decision tree has been grown, most inducers use one out of many pruning methods in order to get rid of internal nodes which do not improve the tree's performance. Most pruning methods traverse the nodes of the decision tree either top-down or bottom-up and perform an error comparison with and without the selected node. If the error is lower or within a certain threshold regarding a particular measure, the node can be deleted without greatly decreasing the accuracy of the whole tree. Some of the more popular pruning methods found in literature are described in the following paragraphs:

Cost-Complexity Pruning

The Cost-complexity pruning method (or weakest link pruning) works in two steps. First, multiple trees are generated by using the training data, starting with the original tree before pruning and ending with the root tree, so every newly generated tree has less nodes than the one before. For every new tree, the subtrees whose

deletion has the lowest negative impact on the error-rate in the predecessor tree are replaced by suitable leaves. In the second step, the tree with the smallest estimation of the generalization error is chosen as the resulting pruned tree (Breiman et al., 1984).

Error-based Pruning

In the Error-based Pruning or EBP, as in most other pruning methods, an estimated error rate is calculated, defined as Maimon and Rokach (2010):

$$\epsilon_{U,B}(T, S) = \epsilon(T, S) + Z_\alpha \sqrt{\frac{\epsilon(T, S)(1 - \epsilon(T, S))}{|S|}} \quad (2.6)$$

After calculating the error rate estimation, all nodes are traversed bottom-up and there are three possibilities for a node. Depending on the lowest estimated error rate of either the node's subtree, its pruned subtree or its subtree starting with the child with the biggest subtree of its own, the method leaves the node as is, prunes it or replaces it with the subtree starting with its biggest child.

Optimal Pruning

In (Bohanec and Bratko, 1994), a method for optimal pruning known as OPT was first presented. It uses dynamic programming with a runtime of $\Theta(|leaves(T)|^2)$. This method was then improved by (Almuallim, 1996) with a more efficient runtime of $(|leaves(T^*)| |internal(T)|)$, with T^* being the smaller, already pruned tree.

Others

Other pruning methods include Pessimistic Pruning (Quinlan, 1993), Minimum Error Pruning (Olaru and Wehenkel, 2003), Reduced Error Pruning (Quinlan, 1987) and MML (Minimum Message Length) (Wallace and Patrick, 1993).

Performance

Similar to the comparison of Splitting Criteria, there have been some studies on the performance on pruning methods such as (Quinlan, 1987), (Mingers, 1989) and (Esposito et al., 1997), but none of the tested methods could be confirmed as being the best overall choice in every case.

2.1.1.3 Inducers

ID3

This inducer is one of the first decision tree algorithms to come into existence. It uses information-gain as splitting criterion and does not implement methods for pruning, handling numeric attributes or missing values (Quinlan, 1986).

C4.5

The C4.5 algorithm was presented by the same author who introduced ID3 and represents an extended version. It uses gain-ratio for splitting and performs error-based pruning. Furthermore, it can handle both missing values as well as numeric attributes (Quinlan, 1993).

CART

Classification and Regression Trees (CART) constructs binary trees using the two-ing criterion. It performs cost-complexity pruning and has the ability to create regression-trees that predict real numbers instead of classes. Users can also provide prior probability distributions and misclassification costs during induction (Breiman et al., 1984).

QUEST

The QUEST (Quick-Unbiased-Efficient-Statistical-Trees) algorithm creates binary decision trees and supports both univariate and multivariate splitting criteria. For each split, an ANOVA F-test, Levene's test (ordinal and continuous attributes)

or Pearson's-chi test (nominal attributes) is performed, calculating the association between the input attributes and the target. It uses ten-fold cross-validation for pruning(Loh and Shih, 1997).

Others

There are many other inducers including CHAID, CAL5, FACT and MARS which can be found in (Lim et al., 2000). Furthermore, there are extensions to the standard paradigm of the decision tree such as the Oblivious Decision Trees (Almuallim, 1996) or the Fuzzy Decision Trees (Janikow, 1998). These extensions address distinct shortcomings or limitations of the classic decision tree like the handling of large data sets (Alsabti et al., 1998), the incremental updating of a decision tree (Freitas and Lavington, 1998) and (Gehrke et al., 1999) or the replication problem described in (Pagallo and Haussler, 1990).

2.1.1.4 Advantages and Disadvantages

Following some of the state-of-the-art literature (Maimon and Rokach, 2010), (Seni and Elder, 2010), the advantages and disadvantages can be summarized as follows:

Advantages

- No assumptions about the space distribution and classifier structure
- Easy to use compared to similarly powerful methods
- Capable of handling missing and faulty data
- Very comprehensible
- Can handle nominal and numeric attributes
- Can represent any discrete-value classifier

Disadvantages

- Sensitive to irrelevant attributes and noise because of the algorithm's greedy nature.
- Most inducers require the target attributes values to be discrete (e.g. ID3, C4.5)
- Does not perform well when facing many complex interactions due to the replication problem (Pagallo and Haussler, 1990)
- Data fragmentation. Training data is reduced with every split.

2.1.1.5 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- CART - <http://www.salford-systems.com/en/products/cart>
- Orange - <http://orange.biolab.si/>
- KNIME - <http://www.knime.org/>

2.1.2 Support Vector Machines

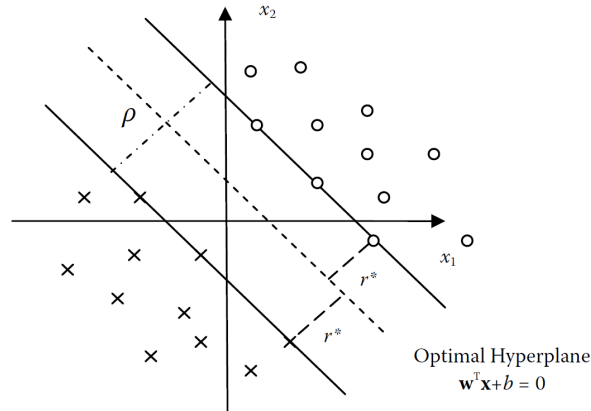


Figure 2.3: Optimal hyperplane for a linearly separable case (Wu and Kumar, 2009)

Support Vector Machines (SVMs) are supervised learning models which can be used for both linear and non-linear classification and regression. They were developed by Vapnik (1995). The method uses hyperplanes to separate data points, searching the hyperplane with the biggest margin to the support vectors, which are the nearest vectors to the hyperplane. By using the maximum margin, the method makes it possible to classify data points correctly, even if they vary from the training data. The hyperplane has to be linear and is only influenced by the nearest data points, ignoring outliers. Obviously, data in real world applications can not always be separated linearly, and for these cases, SVMs provide a technique called the "Kernel-Trick", which basically transforms the feature space into a higher dimension, maybe even infinity, until the data can be separated by a linear hyperplane. The method can be further extended by so called "slack variables", enabling data points to be classified incorrectly, which helps dealing with noisy data and overfitting and also reduces the amount of support vectors necessary. Support Vector Machines perform very well on two-class problems and can be extended for multi-class problems as well, but it is not a method that produces easily comprehensible models. SVMs are and can be used in many different fields such as pattern recognition (Martin et al., 2004), text categorization (Joachims, 2002), facial recognition, bioinformatics, medicine as well as many other data mining tasks (Maimon and Rokach, 2010).

2.1.2.1 Hyperplane Classifiers

Classification by the use of hyperplanes to separate data is essentially not very different from other classification methods with regard to the methodology and the steps involved. The model is trained, using a training set of data and then evaluated on a test set. The goal is, to find a function f , that minimizes the expected error. Support Vector Machines are based on the ideas of Vapnik (1995), stating that a linear function, able to classify most of the data is to be preferred over a highly complex function.

Linear Classifier

In the simplest case, the data points can be separated linearly using a hyperplane. Such a hyperplane is defined as (Hastie et al., 2008):

$$\{x : f(x) = x^T \beta + \beta_0 = 0\} \quad (2.7)$$

where the training data consists of N pairs $(x_1, y_1) \dots (x_N, y_N)$, $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$. β is a unit vector: $\|\beta\| = 1$.

A classification rule induced by $f(x)$ would be (Hastie et al., 2008):

$$G(x) = \text{sign}[x^T \beta + \beta_0] \quad (2.8)$$

In order to find the hyperplane with the largest margin between the two classes, the following optimization problem has to be solved, maximizing the margin between the support vectors and the hyperplane (Hastie et al., 2008):

$$\begin{aligned} & \min_{\beta, \beta_0} \|\beta\| \\ & \text{subject to } y_i(x_i^T \beta + \beta_0) \geq 1, i = 1, \dots, N \end{aligned} \quad (2.9)$$

with $M = 1/\|\beta\|$. For a more detailed explanation of the mathematical implications as well as the solution of the optimization problem, please consult Chapter 4.5.2 of (Hastie et al., 2008).

Kernel Trick

Due to the limitations of a linear classifier, especially in the context of real world applications, Support Vector Machines provide a technique known as "The Kernel Trick" to overcome this problem (Cortes and Vapnik, 1995). Using this technique, the original input space is transformed to a higher dimensional feature space ($\Phi : \mathbb{R}^2 \rightarrow F$). The algorithm applied to find the linear classification of the original input space can be used for the transformed data, fitting a maximum margin hyperplane in the new feature space F . The actual trick is to avoid the real computation of the complex feature space. This works by finding a suitable kernel function of the transformation, which should be a symmetric, positive and semidefinite function (Hastie et al., 2008). No optimal way to find the best kernel function has been developed yet, but active research is taking place in this area (Steinwart, 2003). There are however some popular choices for the kernel function $K(x, x')$ mentioned in literature (Hastie et al., 2008):

- dth-Degree polynomial: $K(x, x') = (1 + \langle x, x' \rangle)^d$
- Radial basis: $K(x, x') = \exp(-\gamma \|x - x'\|^2)$
- Neural network: $K(x, x') = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2)$

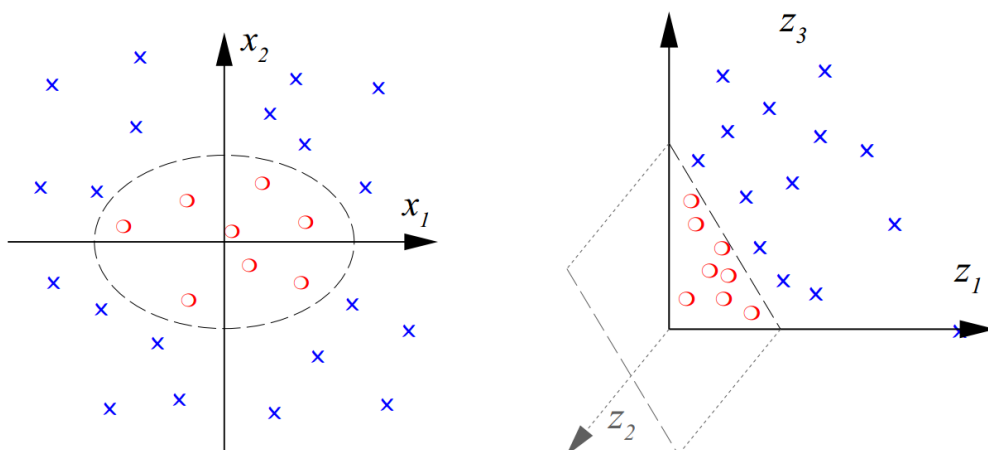


Figure 2.4: Mapping the training data to a higher dimension feature space ($\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$), constructing a separating hyperplane (Scholkopf and Smola, 2001)

Optimization

One of the problems of Support Vector Machines is that the training of a model is highly complex regarding its computation, which means that for large datasets, some models will not be applicable due to the high computation costs. There are however solutions to this problem, such as breaking large optimization problems into smaller ones where each singular problem contains only a certain number of chosen variables from the full model in order to solve the whole problem more efficiently (Wu and Kumar, 2009). Other approaches are proposed and discussed in (Tsang et al., 2005a), (Tsang et al., 2007), (Tsang et al., 2005b) and (Tsang et al., 2006).

Model Selection

In order to optimize the resulting model generated by an SVM, several parameters have to be used, making this method difficult to handle for unexperienced users. Most importantly, the kernel and its parameters have to be chosen, which, as described above, is not a trivial task. Usually, a sequence of models, each using a different set of parameters for the kernel and regularization, is computed and then evaluated. This process can lead to a very high number of computationally costly evaluations. For the evaluation, methods such as cross validation can be used to estimate the generalization error, thus increasing the number of evaluations (Maimon and Rokach, 2010). Another approach based on the the Bayesian evidence framework can be used as well, as described in (Wei, 2003). This method can significantly reduce the number of SVM evaluations needed.

Multiple Classes

Support Vector Machines were originally developed to classify two-class problems which, especially in real-world applications, can be quite limiting. Usually this problem is overcome by extending the SVM to solve multiple two-class problems. For each pair of classes, a classifier is created, with the final classifier being the one that is most dominant as described in (Kressel, 1999).

Soft-Margin

In order to improve the results of SVMs with noisy datasets by making them more robust to data points that are not able to be separated by the hyperplane, (Cortes and Vapnik, 1995) introduced a concept called "slack variables". The basic idea is to include regularization constants in order to control the maximum amount of allowed training errors. More information on controlling the tradeoff between model accuracy and model complexity in regard to Support Vector Machines can be found in (Maimon and Rokach, 2010) and (Chen et al., 2005).

2.1.2.2 Advantages and Disadvantages

Following the state-of-the-art literature (Maimon and Rokach, 2010), (Hastie et al., 2008), the advantages and disadvantages can be listed as follows:

Advantages

- Fast classification
- High precision and accuracy
- Very good generalization ability
- Good performance in real world problems

Disadvantages

- Needs a comparably long time for training computation (selection of the best hyperplane)
- Optimal Kernel has to be found empirically
- Correct preprocessing necessary

2.1.2.3 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- Orange - <http://orange.biolab.si/>
- KNIME - <http://www.knime.org/>
- SVMlight - <http://svmlight.joachims.org>
- LIBSVM - <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>
- simpleSVM - <http://asi.insa-rouen.fr/~gloosli/simpleSVM.html>
- SVQP - <http://leon.bottou.org/projects/svqp>

2.1.3 Bayesian Networks

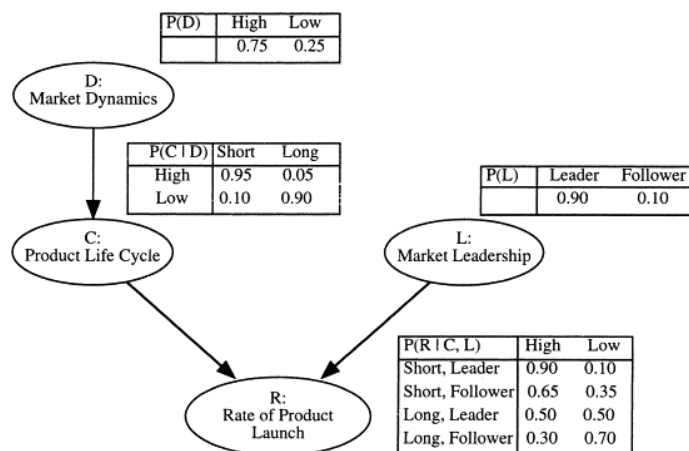


Figure 2.5: Simple example of a Bayesian network representing the impact of different variables on each other. Every variable has a probability table describing its conditional distribution regarding its parents. (Nadkarni and Shenoy, 2001)

Bayesian networks (Pearl, 1988) are a knowledge discovery method belonging to the class of probabilistic graphical models (Whittaker, 1990), (Lauritzen, 1996). The graph describing a Bayesian network is a directed acyclic graph, where nodes represent random variables and arcs form their conditional dependencies. If nodes don't have parents, they describe an unconditional probability distribution. Bayesian networks consist of the graph and of a probability distribution representing the conditional dependencies between the stochastic variables in the model. This method can be used to find information about the relationships between variables as well as for making predictions (Maimon and Rokach, 2010). There are several algorithms used for classification and modeling based on Bayesian networks such as Naïve Bayes, which are used in various fields such as computational biology, bioinformatics, medicine, engineering and image processing, to name just a few. The method's solid and thoroughly studied statistical background as well as its diversity regarding the types of input data make it one of the most promising approaches in the field of knowledge discovery (Maimon and Rokach, 2010). Due to the widespread popularity of the method and its relative simplicity of implementation in some cases, a vast amount of different software implementations is available, ranging from commercial data mining software to open source frameworks implementing different models based on Bayesian networks.

2.1.3.1 Reasoning

Bayesian networks are able to evolve into complete simulation systems which are capable of performing prediction as well as the computation of the conditional distribution of the involved variables, thus explaining relationships (Maimon and Rokach, 2010). According to (Maimon and Rokach, 2010), the most common approaches to perform inference within Bayesian networks are the following:

- Polytrees

If the graphical representation of a Bayesian network is that of a Polytree, which is a directed acyclic graph (DAG) with exactly one linking path between any two vertices, that graph can be divided into two disjoint subtrees at any node. This enables local propagation with increased efficiency.

- Conditioning

The approach of conditioning requires that, a Bayesian network with a graph more complex than a Polytree, but with an instantiated subset of its nodes called a loop cutset (Cooper, 1990), can be reduced to multiple Polytrees, making it possible to apply the Polytree-approach as described above.

- Clustering

The Clustering approach by (Lauritzen and Spiegelhalter, 1990), transforms a Bayesian networks graph into a Polytree structure called a "junction tree" by merging selected attributes. This method allows for local propagation in Bayesian networks consisting of loops, while retaining global consistency throughout the graph.

- Goal-Oriented

This approach is targeted at analyzing which variables inside a Bayesian network are not relevant for computing the posterior probability of the target variable, by querying the probability distribution of that variable (Shachter, 1986).

Other approaches include stochastic simulation (Cheng and Druzdzel, 2000) and Gibbs sampling (Geman and Geman, 1984).

2.1.3.2 Learning

If the task at hand is to learn a Bayesian network from data, the most important and, at the same time, most challenging task is to find and select a suitable model in an efficient way. After the model has been selected, the subsequent critical issue is its validation, which is necessary to evaluate how well the selected model fits the data.

There are several heuristic searching methods, simplifying the problem of model search in Bayesian networks as described in (Cooper and Herskovits, 1992), (Larranaga et al., 1996) and (Singh and Valtorta, 1995) among others. Usually these stochastic and deterministic search procedures aim to make the search space smaller, thus improving the efficiency.

There are several approaches to the model selection in Bayesian networks such as hypothesis testing as described in (Maimon and Rokach, 2010) and (Ramoni and Sebastiani, 2003) the MDL (minimum description length) score and the BIC (Bayesian information criterion) (Lauritzen, 1996) and (Whittaker, 1990).

According to (Maimon and Rokach, 2010), the two most popular approaches used to validate multivariate Bayesian network models are the evaluation of the goodness of fit and the prediction of the network's accuracy. The goodness of fit basically evaluates the accuracy of the model in regard to the source data, summarizing the differences between them. A specific procedure using this approach is known as "blanket residuals" by Ramoni and Sebastiani (2003). The predictive accuracy of a model can be tested using an independent test set. If there is no such test set, cross validation techniques can be used for retraining and testing (Hastie et al., 2008).

2.1.3.3 Data Mining

Naïve Bayes

In the field of supervised classification, the Naïve Bayes algorithm is one of the most popular methods due to its simplicity, comprehensibility and overall performance compared to similar methods (Hand and Yu, 2001).

The most basic version of this algorithm is the Naïve Bayes classifier (NBC) (Duda and Hart, 1973b) and (Langley et al., 1992). The probability distribution over the

class C is defined as (Maimon and Rokach, 2010):

$$p(y_{1k}, \dots, y_{vk} | C_k) = \prod_i p(y_{ik} | C_k) \quad (2.10)$$

During training, the conditional probability distributions of all variables regarding the class are estimated. The classification works by using the posterior probability distribution, assigning the input to the maximum posteriori probability class.

There are extensions to the original NBC such as TAN (Tree Augmented Naïve Bayes), ANB (unrestricted Augmented Naïve Bayes) (Friedman et al., 1997) and l-LDB (l-Limited Dependence Bayesian) (Sahami, 1996).

Dynamic Bayesian networks

Dynamic Bayesian networks (Ghahramani, 1998) are an extension of the classic paradigm of Bayesian networks, tackling the problem that the conditional independencies inside a directed acyclic graph cannot have cycles, thus hindering the model in its ability to represent forward loops (Maimon and Rokach, 2010).

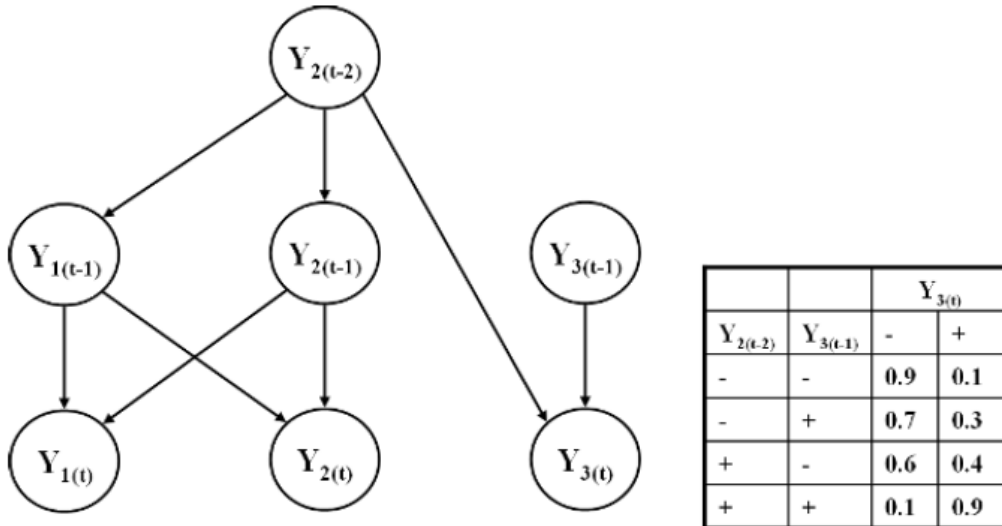


Figure 2.6: Dynamic Bayesian network representing temporal dependency between Y_1, Y_2 and Y_3 (Maimon and Rokach, 2010)

The basic structure of the directed acyclic graph is the same, but dynamic Bayesian networks provide functionality for multivariate time series, feed-forward

loops and feedback within the graphical representation, making them suitable to be applied in areas such as engineering or medicine, where temporal dependencies as well as conditional feedback can be an issue. For more information on the subject of dynamic Bayesian networks, consult (Maimon and Rokach, 2010) and (Friedman et al., 1998).

2.1.3.4 Advantages and Disadvantages

Following some of the state-of-the-art literature (Maimon and Rokach, 2010), the advantages and disadvantages can be summarized as follows:

Advantages

- It is easy to add new data
- Fairly comprehensible and interpretable
- Are able to draw probabilistic inferences on any variable in the data set

Disadvantages

- No universally accepted method to create a network from data
- Problems handling incomplete data
- The quality of the model depends on the prior model.

2.1.3.5 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- Orange - <http://orange.biolab.si/>
- KNIME - <http://www.knime.org/>
- AgenaRisk - <http://www.agenarisk.com/>
- Bayesia - <http://www.bayesia.com/>
- Bayes Server - <http://www.bayesserver.com/>
- HUGIN - <http://www.hugin.com/>
- jBNC - <http://jbnc.sourceforge.net/>
- WINBUGS - <http://www.mrc-bsu.cam.ac.uk/bugs/winbugs/contents.shtml>

2.1.4 Regression Analysis Methods

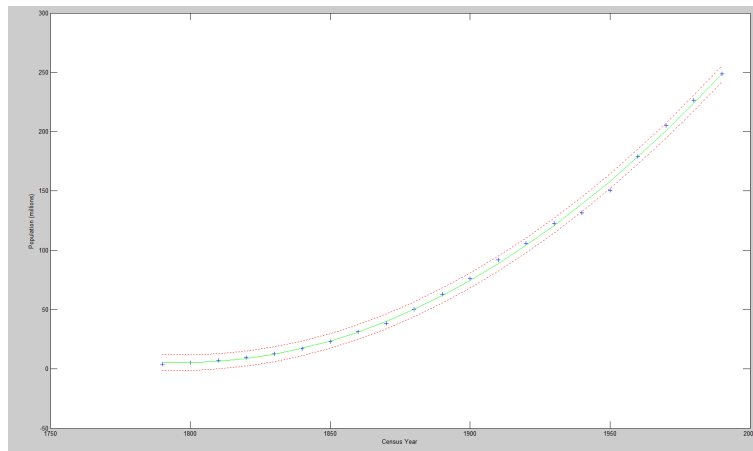


Figure 2.7: Fitting of a regression model using MATLAB’s polyfit function on a demo dataset.

Regression Analysis in statistics describes a collection of methods and techniques that analyze the relationships between dependent and independent variables. Regression models can be used to perform prediction as well as to explore and understand different relationships between the dependent and independent variables. The usual procedure is to determine a function f with a minimized error (also called residuals), where the form and dimensionality of the function are often predetermined by the chosen method, which could be, for example linear- or logistic regression. Due to its long history and solid statistical background, Regression Analysis has been and still is an active area of research, with a vast amount of methods and extensions for modeling a wide variety of datasets. Methods based on Regression Analysis are being used in just about every field that comprises any form of statistics for data mining such as economics, all the natural sciences, engineering and many others. This section provides an overview of a few representative and widely used data mining methods within Regression Analysis based on Chapter 11 in (Maimon and Rokach, 2010).

2.1.4.1 Smoothers

In the context of Regression Analysis, smoothing algorithms are methods aimed to increase the flexibility and robustness of models, especially considering noisy data.

Smoothing Splines

Smoothing Splines are a way to avoid the general knot selection problem of spline-based methods. Regression splines, for example, use a maximal set of knots and a smoothing parameter λ . More information on regression and smoothing splines can be found in (Hastie et al., 2008).

An example of a popular method based on regression splines is MARS (Friedman, 1991).

Lowess

The Locally Weighted Linear Regression Smoother or Lowess (Cleveland, 1979) is a highly popular smoothing method for models with a single predictor. The basic idea behind it is, for each data point, to take x_i as the center of a window that includes all values around x_i and assign a weight to each value according to their distance from x_i , with x_i having the maximal weight. The smoothed value for y in regard to x_i is then calculated by using the weighted least squares method. These fitted values of y then form the Lowess curve.

GAM

The General Additive Model (GAM) described by Hastie and Tibshirani (1990), is a statistical model used to characterize nonlinear regression effects. It is one of the most popular strategies found in literature for dealing with the smoothing of multiple predictors. The mean function for p predictors can be defined as (Maimon and Rokach, 2010):

$$\bar{y}|x = \alpha + \sum_{j=1}^p f_j(x_j) \quad (2.11)$$

The GAM, same as the GLM (generalized linear model) provides the opportunity to use link functions and disturbance distributions (Maimon and Rokach, 2010). There are several fitting algorithms using backfitting as described in (Hastie and Tibshirani, 1990), but also other approaches such as B-splines (Wood, 2004) using penalized regression.

2.1.4.2 CART

Classification and Regression Trees (Breiman et al., 1984) is one of the most popular data mining methods for both classification and regression. The classification-part of this method is described in 2.1.1.

CART grows a regression-tree by partitioning the data using multiple predictors (attributes). For each predictor, a splitting point is calculated, which minimizes heterogeneity in the resulting partitions using the least sum of squared errors. This splitting process is repeated for both resulting partitions until there is no splitting point available or a splitting criterion is reached. In the resulting tree, the whole dataset resides in the root node while internal nodes contain intermediate partition steps and terminate nodes are the final partitions of the dataset in regard to the considered predictors.

CART uses cost-complexity pruning to prune the tree after growing it as is described in 2.1.1, in order to reduce its complexity.

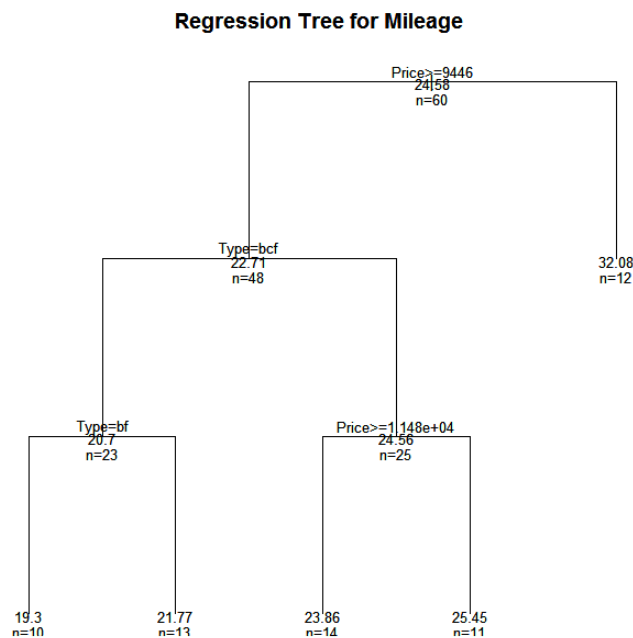


Figure 2.8: Regression tree describing predicted mileage of cars.
Image source: <http://www.statmethods.net/advstats/cart.html>.

2.1.4.3 Bagging & Random Forests

Bagging

Bagging or Bootstrap aggregating is an improvement tool for machine learning methods developed by Breiman (1996). This process is based on the principle that so called bootstrap-samples (i.e. samples that occur in all of the training sets) have at least some dependency between each other. Bagging creates new, smaller training sets from the original ones by sampling and replacement, creating some of the bootstrap samples mentioned above. These newly generated training models are then fitted and averaged, leading to an improvement of unstable procedures such as CART (Breiman, 1996).

Random Forests

Random Forests (Breiman, 2001) is an ensemble method for both classification and regression. It combines the concepts of Bagging with decision trees such as CART. In the first step, a large number of trees is created. Then, for each split in the tree, $m \ll M$ features are chosen, where M is the number of original features and then, averaged over all trees, the split is calculated. In this case, each one of the trees gets a "vote" on the question which value (in case of regression) or class (in case of classification) should be chosen. The trees do not get pruned. This method generally provides a higher accuracy than models like CART, but loses some of its comprehensibility (Maimon and Rokach, 2010).

2.1.4.4 Advantages and Disadvantages

Considering some of the state-of-the-art literature (Maimon and Rokach, 2010), the advantages and disadvantages can be summarized as follows:

Advantages

- Solid theoretical background, well understood, lots of examples
- Flexible regarding numeric and categorical inputs
- Does not necessarily require a model

Disadvantages

- Sensitive to outliers
- Multicollinearity between variables can be a problem
- Only relationships are discovered, no underlying causality

2.1.4.5 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- Orange - <http://orange.biolab.si/>
- KNIME - <http://www.knime.org/>
- R - <http://www.r-project.org/>
- SAS/STAT - <https://www.sas.com>
- DTREG - <http://www.dtrek.com/>
- MARS - <http://www.salford-systems.com/en/products/mars>
- MATLAB - <http://www.mathworks.com>

2.2 Unsupervised Methods

Unsupervised learning methods deal with the problem of finding structure in data and separate it from noise, but contrary to supervised methods, they do so without errors or feedback. Because of this circumstance, it is very hard to evaluate the performance or validity of an unsupervised learning method and, without an objective measure of effectiveness, it becomes a matter of opinion which algorithm is the best for different cases (Hastie et al., 2008). The most popular methods include the clustering methods, association rules and link analysis as mentioned below.

2.2.1 Clustering Algorithms

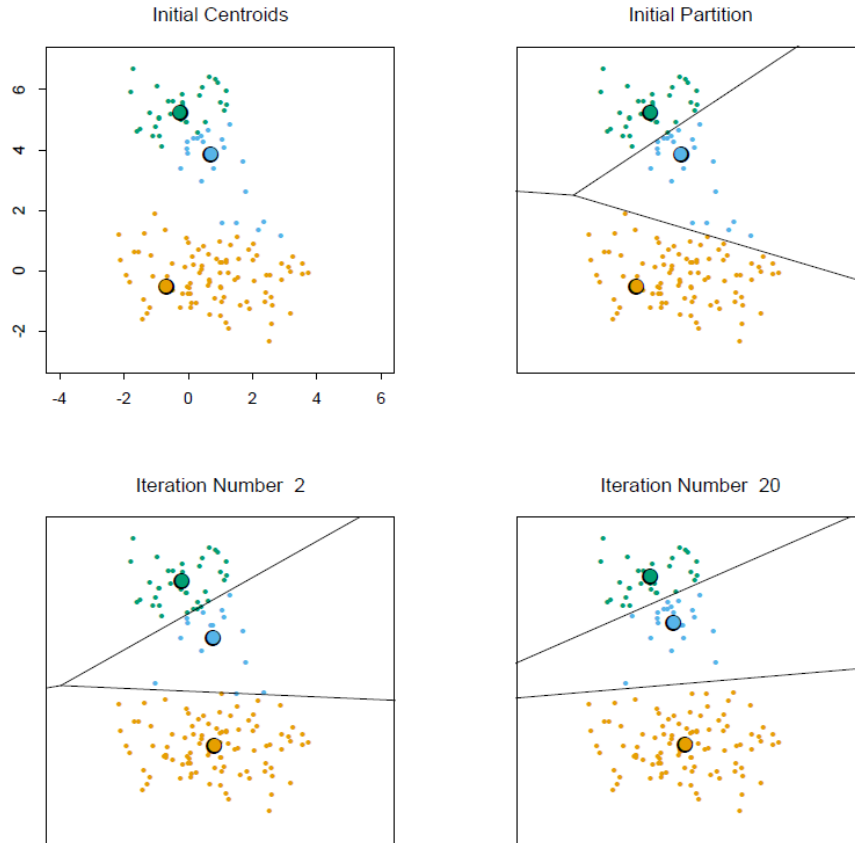


Figure 2.9: Iterations of the k-means algorithm on a simulated dataset by (Hastie et al., 2008)

Clustering algorithms relate to unsupervised learning like classification algorithms do to supervised learning. They aim to perform segmentation between objects using either distance measures or similarity / dissimilarity measures to do so. Due to the wide field of application, there have been approaches in different directions in regard to the general process behind the algorithms. This section adapts the structure of (Maimon and Rokach, 2010), because it captures the diversity of methods in clustering analysis very well. The goal behind clustering algorithms is to find or discover new categories within unlabeled data, by exploiting the fact, that similar objects tend to cluster together. Objects in a clustering context only belong to one cluster at a time, making it possible to segment whole groups from each other while providing good comprehensibility in the process. Clustering analysis is used

in a lot of different fields such as mathematics, biology, genetics, manufacturing, computer science and medicine. The most popular method in the field is the k-means algorithm due to its simplicity both in the handling and implementation and its very good performance on large datasets (Wu and Kumar, 2009).

2.2.1.1 Distance

As mentioned in the introduction of this section, there are two main approaches to segmenting data in cluster analysis: measuring the distance of data points and measuring their similarity to each other. First, a short overview of the most widely used distance measures mentioned in (Maimon and Rokach, 2010) is provided.

The Minkowski-measure aims at calculating the distance between objects of a numeric nature by determining the weighted Minkowski metric with $w_i \in [0, \infty)$ (Han, 2005):

$$d(x_i, x_j) = (w_1|x_{i1} - x_{j1}|^g + \dots + w_p|x_{ip} - x_{jp}|^g)^{1/g} \quad (2.12)$$

For calculating the distance between nominal attributes, there are two approaches. The first one is to match them by using (Maimon and Rokach, 2010)

$$d(x_i, x_j) = \frac{p - m}{p} \quad (2.13)$$

with p being the number of attributes and m the amount of matches. The second approach uses binary attributes for each state of the nominal attributes with q and t being the amount of attributes equal to 1 and 0 for both objects and s and r describing the attributes unequal for both. (Maimon and Rokach, 2010)

$$d(x_i, x_j) = \frac{r + s}{q + r + s + t} \quad (2.14)$$

Ordinal attributes can be handled by mapping their sequence to $[0, 1]$. With, according to Maimon and Rokach (2010), $z_{i,n}$ being the standardized attribute value of object i , $r_{i,n}$ the attribute value of i before standardization and M_n the upper limit of the attribute domain:

$$z_{i,n} = \frac{r_{i,n} - 1}{M_n - 1} \quad (2.15)$$

More detailed information on distance measures in the context of clustering analysis can be found in (Hastie et al., 2008) and (Maimon and Rokach, 2010).

2.2.1.2 Similarity

Similarity measures are the other concept used to segment objects and measure their distance. They aim to calculate the similarity between vectors using different measures, such as the ones mentioned in (Maimon and Rokach, 2010).

The cosine measure is useful, if the angle between vectors is of importance regarding their similarity. Maimon and Rokach (2010) suggest the normalized inner product between two vectors for a similarity measure:

$$s(x_i, x_j) = \frac{x_i^T \cdot x_j}{\|x_i\| \cdot \|x_j\|} \quad (2.16)$$

The Pearson correlation with \bar{x}_i as the average feature value of x over all dimensions (Maimon and Rokach, 2010):

$$s(x_i, x_j) = \frac{(x_i - \bar{x}_i)^T \cdot (x_j - \bar{x}_j)}{\|x_i - \bar{x}_i\| \cdot \|x_j - \bar{x}_j\|} \quad (2.17)$$

Another useful measure is the extended Jaccard Measure (Strehl and Ghosh, 2000), defined as (Maimon and Rokach, 2010):

$$s(x_i, x_j) = \frac{x_i^T \cdot x_j}{\|x_i\|^2 + \|x_j\|^2 - x_i^T \cdot x_j} \quad (2.18)$$

More detailed information on the topic of similarity measures in the context of clustering analysis can be found in (Hastie et al., 2008) and (Maimon and Rokach, 2010).

2.2.1.3 Evaluation

In unsupervised learning, it is very hard to evaluate the performance or the effectiveness of different methods, but there are some approaches that use internal and external information of evaluated models in order to measure their performance. Internal criteria include the SSE (Sum of Squared Error) criterion, which is the most popular one using the well known approach of calculating the minimal error for dif-

ferent models. Another internal criterion known as the scatter criterion as described in (Maimon and Rokach, 2010), tries to measure the scatter between clusters, together with the Condorcet's criterion and its extension, the C-Criterion (Fortier and Solomon, 1996) as well as the Category Utility Metric (Gluck and Corter, 1985). There are also external measures that try to evaluate the performance of clustering algorithms by including external information like templates for predetermined classification. (Maimon and Rokach, 2010) mentions three external evaluation measures, one of them being the Mutual Information Based Measure (Strehl and Ghosh, 2000), which counts instances that are in the same class in both the original clustering and a predefined classification. Another tool known as the Precision Recall Measure, which queries the cluster for a class and measures the results based on the amount of correctly retrieved instances. The third approach mentioned in (Maimon and Rokach, 2010) is the Rand Index (Rand, 1971), which compares two clustering structures in a rather simple way.

2.2.1.4 Methods

This section describes a few of the more popular clustering methods structured according to their super-categories, based on the overview given in (Maimon and Rokach, 2010). Another source of information on different clustering algorithms is chapter 14 of (Hastie et al., 2008).

Hierarchical

Hierarchical clustering algorithms use recursive partitioning with either an agglomerative (merge clusters) or a divisive (divide clusters) approach, performed either top-down or bottom-up (Maimon and Rokach, 2010). According to Guha et al. (1998), the strengths of hierarchical approaches lie in their versatility regarding data sets and their ability to provide multiple usable partitions. The weaknesses are their non-linear scaling and the missing back-tracking capability of hierarchical methods. There are three types of hierarchical algorithms as described in (Maimon and Rokach, 2010), namely Single-link clustering (Sokal and Sneath, 1963), Complete-link clustering (King, 1967) and Average-link clustering (Ward, 1963) and (Murtagh, 1983).

Partitioning

The basic idea behind partitioning methods is to create an initial partitioning, either randomly or with some kind of constant structure and then, by moving instances from one cluster to another and simultaneously calculating the difference in error, to improve the overall structure of the cluster. There are, however, both error minimization approaches and graph-theoretic approaches (Zahn, 1971) and (Urquhart, 1982). This survey focuses on error minimizing algorithms, such as k-means.

Error minimization methods are the most widely used algorithms in cluster analysis due to their intuitive handling and simplicity, with the most popular one being the k-means algorithm which uses a squared error criterion. Compared to other clustering algorithms, the k-means algorithm is simple to use and implement while still being very fast, even in the case of large data sets. There are, however, also disadvantages such as a lack of versatility, sensitivity to outliers, noise and difficulties regarding the initial partitioning. (Maimon and Rokach, 2010)

After the initial partitioning, the k-means algorithm starts iterating by moving each instance to its nearest cluster and then recalculating the center of each cluster as defined in (Maimon and Rokach, 2010), with N_k being the number of instances in cluster k :

$$\mu_k = \frac{1}{N_k} \sum_{q=1}^{N_k} x_q \quad (2.19)$$

After this step, the partitioning error is calculated until it does not improve anymore or another stopping criterion is met. An extension to k-means, known as the k-prototypes algorithm, is able to handle non-numeric data as well.

Another partitioning algorithm is PAM (partition around medoids) by Kaufman and Rousseeuw (1987). While it is based on the same principles as k-means, this algorithm uses another value as the center of the clusters. Instead of the mean of all instances, the most centric instance is chosen as the center, making PAM more robust, but on the other hand more expensive to calculate.

Density

Density-based methods segment data based on probability distribution as described in (Banfield and Raftery, 1993). They grow the clusters until the neighborhood of a group reached a certain amount (threshold) of objects. Examples of algorithms based on density are the EM (Expectation Maximization Algorithm) by (Dempster et al., 1977), AUTOCLASS by (Cheeseman and Stutz, 1996), SNOB by (Wallace et al., 1994) and MCLUST by (Fraley, 1998).

Others

According to Maimon and Rokach (2010), other methods of clustering analysis include model-based clustering methods such as COBWEB, as well as algorithms based on neural networks and fuzzy logic. Furthermore, there are approaches using evolutionary algorithms, particularly genetic algorithms and simulated annealing methods.

2.2.1.5 Advantages and Disadvantages

There are many different clustering algorithms in various categories, depending on how they work. Each category and sometimes even each individual algorithm has a very specific set of strengths and weaknesses, making it impossible to list global advantages and disadvantages. For the description of the different methods mentioned in this section, please consult 2.2.1.4.

The most popular algorithm in the field of clustering analysis is undoubtedly the k-means method (Wu and Kumar, 2009). For large datasets, k-means and variations of it perform very well, compared to other algorithms, but if large data sets and performance are not an issue, algorithms based on the hierarchical, evolutionary and simulated annealing approaches outperform k-means in almost all cases (Al-Sultana and Khan, 1996). There are however theoretical and practical approaches to improve the performance of these algorithms when used on large datasets as described in (Bentley and Friedman, 1978) and (Can, 1993), as well as techniques for dealing with the problem of having to know the amount of clusters beforehand (Tibshirani et al., 2001).

2.2.1.6 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- Orange - <http://orange.biolab.si/>
- ELKI - <http://elki.dbs.ifi.lmu.de/>
- KNIME - <http://www.knime.org/>
- SAS - <https://www.sas.com>
- MATLAB - <http://www.mathworks.com/>

2.2.2 Link Analysis

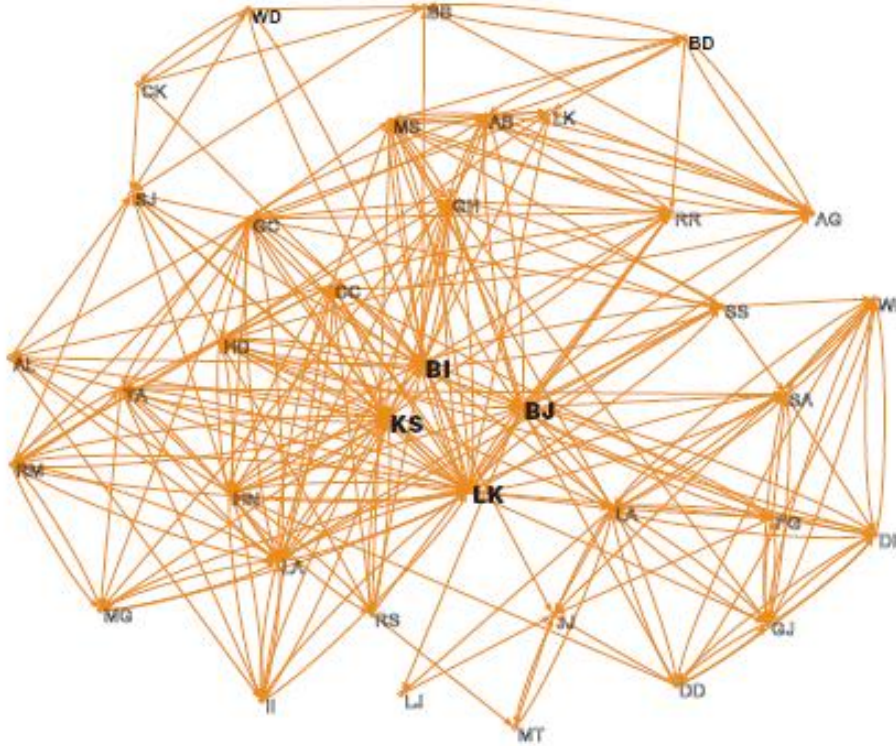


Figure 2.10: Network visualization of knowledge relationships between scientists in a study performed by IBM (Cross et al., 2002)

Link analysis or network analysis is a technique which uses concepts of graph theory to find relationships between nodes in a graph. These nodes, or entities are interconnected, their edges representing certain relationships, depending on the data context. This survey of applications and techniques used in the field uses the structure and overview provided in (Maimon and Rokach, 2010), focusing on SNA - Social Network Analysis, Search Engine Optimization and Law enforcement methods based on link analysis.

These are, however, just some of the areas where link analysis is a popular tool for data mining and analysis. Link analysis and combinations of it with other knowledge discovery methods, as described in (Taskar, 2001) and (Neville et al., 2003) using clustering or classification techniques as described in (Chakrabarti et al., 1998) and (Lu and Getoor, 2003) have been successfully applied in many areas such as medicine, sociology, psychology, market research and computer science.

2.2.2.1 SNA

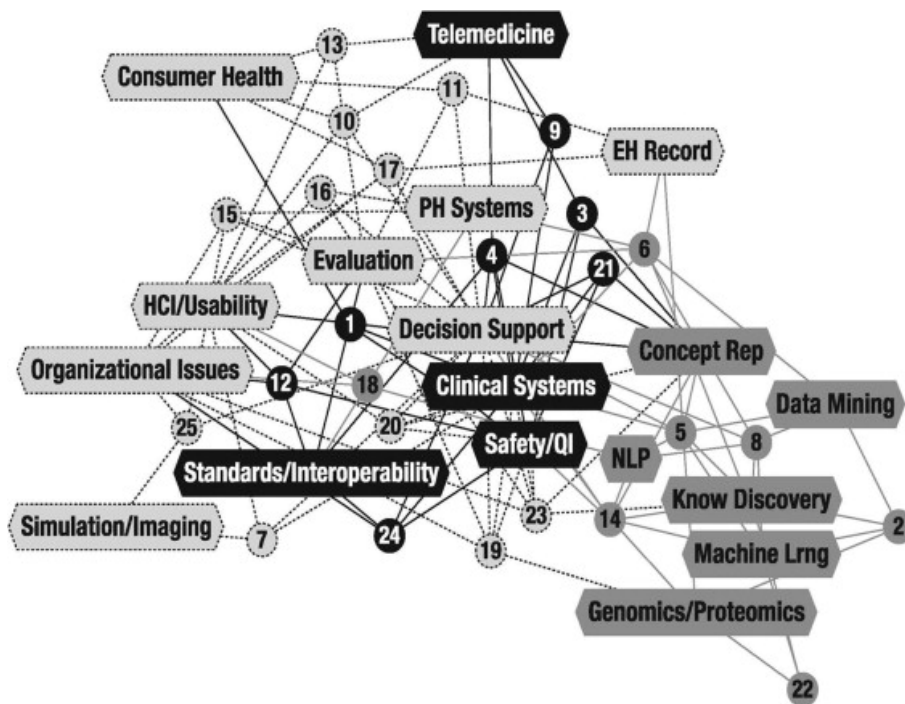


Figure 2.11: Social Network representing faculty members and their areas of expertise within a department of biomedical informatics by Merrill and Hripcsak (2008)

Social Network Analysis (Wasserman et al., 1994) and (Hanneman and Riddle, 2005) deals with the problem of analyzing the relationships between people and groups of people, trying to discover useful information such as which people are the most connected, have the highest influence or how information spreads among groups of a network and in the entire network. Another goal is to compare different networks, or parts of networks to each other in order to find similarities and anomalies, thus making it possible to understand how structure and relationship influence groups of people as well as individuals. For measuring a node's influence or power, its centrality is a key factor. (Maimon and Rokach, 2010) suggests four methods to measure this property in a quantitative way:

- Degree

The degree of a node is determined by counting the amount of nodes linked to it. A node with a high degree can be regarded as "popular" within the network, improving its ability to influence a large amount of nodes.

- Closeness

Closeness describes the distance or amount of nodes to pass from one node to each other node. On average, nodes with high closeness, intuitively, have shorter paths to other nodes.

- Betweenness

The betweenness of a node describes how many shortest paths from all nodes to all other nodes lead through the considered node. High betweenness suggests a high amount of control within the network.

- Cutpoints

Cutpoints, as their name suggests, refer to nodes that, if deleted would cut off parts of the network, rendering these parts disconnected from the network. It measures the amount of "cut-off" networks that would emerge, were the node to be removed.

Another important concept regarding SNA are "cliques", which are groups with high interconnection. According to Maimon and Rokach (2010), they can be defined effectively as follows, using their strictness as basis:

- All nodes inside the clique are connected to all other nodes within.
- K -plexes, where nodes inside the clique have to be connected to at least $N - K$ other nodes, with N being the number of nodes.
- K -cores, where the nodes in a clique have to be connected to at least K other nodes.

In order to discover complex relations and properties between nodes within a social network, Maimon and Rokach (2010) suggest the concept of equivalence, which makes it possible to measure similarity both inside a network as well as outside, comparing whole network structures or groups. According to Maimon and Rokach

(2010) structural equivalence exists, where two nodes are equivalent. This is the case if they have exactly the same nodes connected to them. It is measured by the degree of nodes overlapping between two networks. Regular equivalence between two nodes is given, if their linked nodes are regular equivalents.

2.2.2.2 Search Engine Optimization



Figure 2.12: Visualization of the top ranked image search results for the keyword "Mona Lisa" using PageRank by (Jing and Baluja, 2008)

The Internet is a huge area with vast amounts of data of different kinds available to the public. Additionally, every user has the possibility to extend this network, generating content and linking it to other content that is already present. These circumstances make search engines an essential part of navigating the Internet efficiently. An intuitive way of doing this, would be to search for keywords and return every document related to the keyword to the user, but due to the amount of data available and the possibility of abuse, the search results have to be ranked by some measure of importance in order to provide the user with the results he most likely

was looking for in the first place. According to Maimon and Rokach (2010), two main approaches to determine this ranking can be applied, namely the highly popular PageRank algorithm (Page et al., 1998) and Kleinberg's Hubs and Authorities (Kleinberg, 1999).

PageRank is among the best known knowledge discovery algorithms, because of Google's success using it in their search engine. The algorithm determines a score for every considered website, depending on the amount of pages linked to it and the score of these sites. For outgoing links of a website, the site's score is divided evenly among all outgoing links. In (Page et al., 1998), the calculation of a page's rank is defined as:

$$R(u) = c \sum_{v \in B_u} \frac{R'(v)}{N_v} + cE(u) \quad (2.20)$$

Where u is a web page, $E(u)$ is a certain source of ranks, N_u is the number of links from u , B_u are the pages linked to u and c is a factor for normalization. $E(u)$ counteracts so called "sinks", referring to pages which are linked to by other pages, but do not link anywhere themselves. Usually the PageRank algorithm is used together with filtering techniques such as keyword-searches in the context of search engines.

Although the Algorithm of Kleinberg (Kleinberg, 1999), show certain similarities, it is distinctly different from PageRank, with regard to the assignment of scores, especially concerning hubs and authorities. One of the differences is, that Kleinberg's method includes a keyword search in the calculation of the page's rank, whereas in the case of PageRank, the search is unrelated to the page's score, into the calculation of the page's rank. (Maimon and Rokach, 2010) defines hubs as pages pointing to many other pages, based on a topic and authorities as pages, which are pointed to by a large amount of other pages. The method calculates two different scores for hubs and authorities for each page, which is a technique aimed to identify "real" authorities on certain topics, which are usually what users are looking for.

2.2.2.3 Law enforcement methods

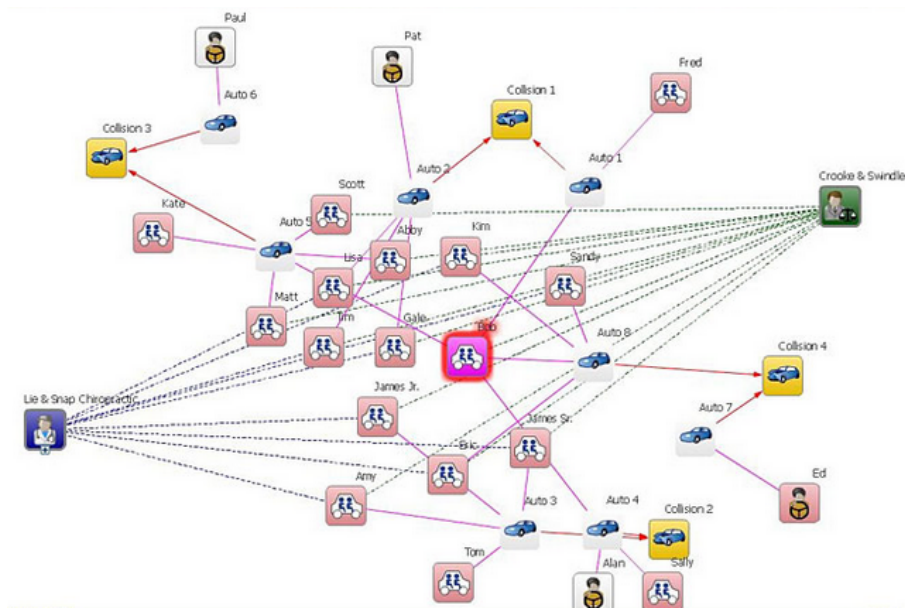


Figure 2.13: Example visualization of a suspicious fraudulent network using the SAS Fraud Framework for Insurance, Source: <http://www.sas.com>

Methods of link analysis help users within the field of law enforcement by visualizing networks containing factors of a crime such as suspects and locations, thus enabling them to combine these relationships with external data about the entities. These external sources can be social networks, phone books or information about money transactions, leading to causal relationships between nodes of a network, and thus permitting an understanding about motives and collaborations in crimes.

The basic concepts of link analysis have been heavily used by law enforcement agencies, even before the introduction of sophisticated software systems such as FAIS - Fincen Artificial Intelligence System (Senator et al., 1995). The FAIS is based on data from money transactions, providing an analysis and visualization of a network of transactions between different entities. Fraud detection systems often exploit the fact that, although individual cases of suspicious behavior may not be related to a crime, multiple suspicious, interconnected nodes with different relations to each other can draw a much clearer picture of underlying criminal behavior.

Another concept in this field is the template approach, which compares the structure of groups within networks to the structure of already proven crimes as described in (Ruspini et al., 2004) and (Daniel Fu, 2003).

2.2.2.4 Advantages and Disadvantages

Following some of the state-of-the-art literature (Maimon and Rokach, 2010), the advantages and disadvantages can be summed up as follows:

Advantages

- Comprehensible
- A lot of publicly available data
- Very good for analyzing and visualizing relationships between entities

Disadvantages

- Often only visualization
- Limited to analyzing relationships

2.2.2.5 Existing Software

- Maltego - <http://www.paterva.com/web6/>
- Network Workbench - <http://nwb.cns.iu.edu/>
- R - <http://cran.us.r-project.org/web/packages/sna/index.html>
- SPSS - <http://www-01.ibm.com/software/analytics/spss/>
- SAS - <https://www.sas.com>
- NetMiner - <http://www.netminer.com/index.php>
- Orgnet - <http://www.orgnet.com/>

2.2.3 Association Rules

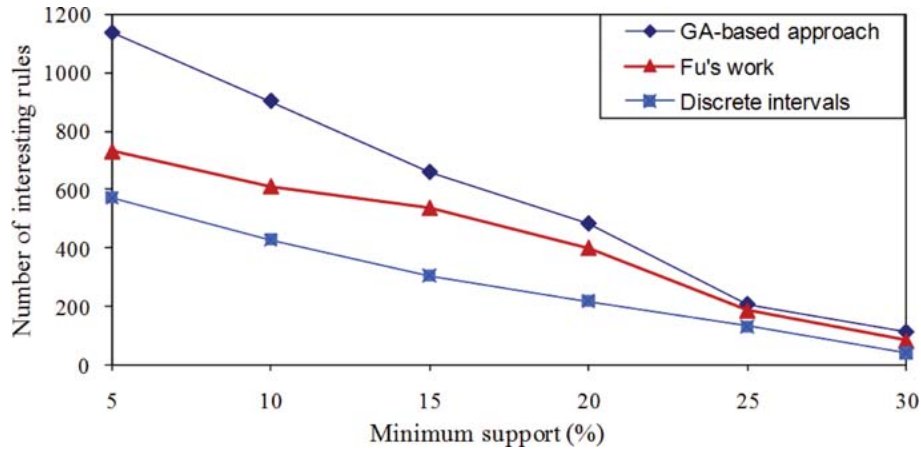


Figure 2.14: Number of interesting association rules from (Kianmehr et al., 2011), with different minimum-support values

Association rule mining (Agrawal and Srikant, 1994) is a popular knowledge discovery method for discovering relations between variables in data sets. They emerged as an approach to help solve the market-basket analysis problem, which basically tries to describe which products consumers usually buy together such as "Salami" and "Bread", by finding strong rules between transaction record data points.

The general method of association rule mining can, of course, be applied to a much wider variety of applications such as linguistics (Aumann and Lindell, 1999), medicine (Gamberger et al., 1999), text mining (Brin et al., 1997) and (Delgado et al., 2002), telecommunication (Mannila et al., 1997) and many others. The most prominent and widely used algorithm in this field is the Apriori algorithm by Agrawal and Srikant (1994).

2.2.3.1 Apriori

The introduction of the Apriori algorithm by (Agrawal and Srikant, 1994) proved to be a large step for data mining in general, due to its easy implementation and fundamental approach, Apriori had and still has a tremendous impact on the world of knowledge discovery (Wu and Kumar, 2009).

The algorithm, same as many methods based on Apriori, works in a two-step process. First, so called "frequent item sets" are gathered from the database. These item sets distinct themselves by satisfying a given support threshold. The support s , in this case, relates to the fraction of item sets which contain the item set at hand within the database. Searching for frequent items happens in multiple passes, where in each pass, the algorithm tries to find new frequent items and looks for new item sets above the support threshold. This is repeated until no further frequent items can be found.

The next step in the Apriori method is to generate association rules from the found frequent item sets, (Wu and Kumar, 2009) describes this as follows: Enumerate all nonempty subsets of all frequent item sets f , then, for every subset a of f , create a rule $a \Rightarrow (f - a)$, as long as the ratio of $support(f)$ to $support(a)$ is above a given confidence value. The confidence in this case is defined by $c(a \Rightarrow f) = \frac{support(a \cup f)}{support(a)}$ and describes the estimated probability for the rule to hold within all item sets. (Wu and Kumar, 2009) further states that $c(\hat{a} \Rightarrow (f - \hat{a})) \leq c(a \Rightarrow (f - a))$, with $\hat{a} \subset a$, which means that for $(f - a) \Rightarrow a$ to hold, all $(f - \hat{a} \Rightarrow \hat{a})$ must hold as well. Relating to this property, (Wu and Kumar, 2009) provides a comprehensible pseudo code, explaining the method Apriori uses to generate association rules as represented in 2.15.

```

 $H_1 = \emptyset$  //Initialize
foreach frequent  $k$ -itemset  $f_k, k \geq 2$  do begin
   $A = (k - 1)$ -itemsets  $a_{k-1}$  such that  $a_{k-1} \subset f_k$ ;
  foreach  $a_{k-1} \in A$  do begin
     $conf = \text{support}(f_k) / \text{support}(a_{k-1})$ ;
    if ( $conf \geq \text{minconf}$ ) then begin
      output the rule  $a_{k-1} \Rightarrow (f_k - a_{k-1})$ 
        with confidence =  $conf$  and support =  $\text{support}(f_k)$ ;
      add  $(f_k - a_{k-1})$  to  $H_1$ ;
    end
  end
  call ap-genrules( $f_k, H_1$ );
end

Procedure ap-genrules( $f_k$ : frequent  $k$ -itemset,  $H_m$ : set of  $m$ -item
  consequents)
if ( $k > m + 1$ ) then begin
   $H_{m+1} = \text{apriori-gen}(H_m)$ ;
  foreach  $h_{m+1} \in H_{m+1}$  do begin
     $conf = \text{support}(f_k) / \text{support}(f_k - h_{m+1})$ ;
    if ( $conf \geq \text{minconf}$ ) then
      output the rule  $f_k - h_{m+1} \Rightarrow h_{m+1}$ 
        with confidence =  $conf$  and support =  $\text{support}(f_k)$ ;
    else
      delete  $h_{m+1}$  from  $H_{m+1}$ ;
    end
  call ap-genrules( $f_k, H_{m+1}$ );
end

```

Figure 2.15: Pseudo code of Apriori's association rule generation by Wu and Kumar (2009)

2.2.3.2 Extensions

The Apriori approach, using confidence and support parameters, can often lead to misleading, useless and trivial results within the generated rules. There are, however, several extensions to the general concept of association rule mining that are able to improve this behavior, such as alternative ways of evaluating rules, different representations of the results and general additional filtering and limiting of the search space. This section provides an overview of available techniques used to improve the results created by association rule mining, based on the methods mentioned in (Maimon and Rokach, 2010).

One approach to achieve this is to improve the evaluation of generated rules, so

that only the "interesting" rules are returned. This is important because a lot of the rules generated by association rule mining can be fairly trivial at times. There are methods focused on measuring the information content, such as the J-measure (Smyth and Goodman, 1992) and statistical methods such as the one described in (Piatetsky-Shapiro, 1991). For a comparison of different methods using the approach of rule evaluation improvement, (Maimon and Rokach, 2010) refers to (Hilderman and Hamilton, 2001).

Another popular approach is to filter the results by using external knowledge from experts, restricting resulting rules by a predefined set of uninteresting or trivial rules (Dong and Li, 1998) and (Padmanabhan and Tuzhilin, 1998). This way, only rules that are different from the ones that are already known are returned, removing a lot of clutter from the resulting rule set.

Algorithms such as Close (Pasquier et al., 1999), Charm (Zaki, 2000) and Closet+ (Wang et al., 2003) are aimed to reduce the amount of rules, which are returned using the concept of condensed representation (Mannila and Toivonen, 1996). Another approach in this direction is that of (Webb, 2000), using optimistic pruning to compress the result set.

All of the approaches mentioned so far rely on an Apriori basis to generate rules, but there are approaches which do not use the traditional support / confidence framework, because it does not make sense in every case regarding the underlying data sets. (Maimon and Rokach, 2010) mentions approaches by (Brin et al., 1997), (Cohen et al., 2000) and (Aumann and Lindell, 1999) among others.

2.2.3.3 Advantages and Disadvantages

Following some of the state-of-the-art literature (Maimon and Rokach, 2010), (Wu and Kumar, 2009), the advantages and disadvantages can be summarized as follows:

Advantages

- Easy to implement
- Widely used with a lot of documentation and examples
- Easy to implement

Disadvantages

- Computationally complex for large datasets
- Always needs result-filtering as Apriori finds all rules, regardless of their value of interest

2.2.3.4 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- Orange - <http://orange.biolab.si/>
- Christian Borgelt's implementation - <http://www.borgelt.net/apriori.html>
- KNIME - <http://www.knime.org/>
- ARMiner - <http://www.cs.umb.edu/laur/ARMiner/>
- SPMF - <http://www.philippe-fournier-viger.com/spmf/>

2.2.4 Constraint-based Methods

This section provides an overview of methods used for data mining with constraint-based methods using inductive querying, based on Chapter 17 in (Maimon and Rokach, 2010). The basic idea behind these methods is to provide a framework for KDD, transforming the knowledge discovery process into an extended, controlled query-based one using approaches such as IDB, Inductive Data Base by (Imielinski and Mannila, 1996). IDB's are databases that do not only contain data, as is the usual case, but also patterns such as classifiers or sequences with the goal to evaluate theories based on these patterns using constraints via querying. According to Maimon and Rokach (2010), inductive querying can be abstracted using the model introduced in (Mannila and Toivonen, 1997), where, given a language L , a database D and a constraint C , a theory is formulated as $Th(D, L, C) = \{\varphi \in L \mid C(\varphi, D) = true\}$, with φ describing a pattern in D . In this case, C can be any constraint such as, in more popular cases in this field, the frequency, for example when selecting frequent item sets like in association rule mining, which is described in 2.2.3.

This general approach can be extended by the use of different constraints as described in (Srikant et al., 1997), (Garofalakis et al., 1999) and (Pasquier et al., 1999), or a more general approach as described in (Ng et al., 1998), regarding anti-monotonicity and non anti-monotonicity, which, same as in (Maimon and Rokach, 2010), will be used to categorize the different methods in the following sections.

Due to the fact, that depending on the complexity of the used constraints, the performance of inductive querying on large datasets can be very complex in regard to computation, another concept that has become very popular in the field of constraint-based data mining is that of condensed representation. This method was introduced by Mannila and Toivonen (1996) and extended in works such as (Jeudy et al., 2002). The basic idea behind condensed representation is to come up with properties or rules, so that the overall number of patterns and items, which would normally have to be tested is reduced improving the comprehensibility and usability of the results as well as the computational complexity of the task. Examples for successful application of this concept are, among others (Bayardo Jr., 1998), (Pasquier et al., 1999), (Boulicaut et al., 2003), (Boulicaut et al., 2000), (Bastide et al., 2000) and (Calders and Goethals, 2002).

2.2.4.1 Anti-monotonic

Anti-monotonic constraints are constraints C_{am} , that, if they hold for an itemset S , they also hold for all subsets of S , S' . With this property being active, large amounts of data can be pruned from the data set, leading to a reduction of the search space, even if only a single one of the subsets of an itemset does not satisfy the constraint C_{am} . Another concept in this context is that of the borders (Mannila et al., 1997) of such constraints, defined by Maimon and Rokach (2010) as $Bd^+(C_{am})$ being the maximal collection of itemsets S satisfying C_{am} and $Bd^-(C_{am})$ being the minimal collection of itemsets not satisfying C_{am} .

A number of algorithms such as the Max-Miner algorithm by Bayardo Jr. (1998) aimed to compute the positive borders. Probably the most influential work in the field regarding algorithms has been carried out by Mannila et al. (1997), introducing the levelwise algorithm. This type of algorithm repeats steps exploiting the anti-monotonic property of being able to prune subsets which are not interesting for the particular search, while searching for new candidates on the patterns in the data. A concrete example of an algorithm based on this principle is the popular Apriori algorithm by Agrawal et al. (1996).

2.2.4.2 Non anti-monotonic

Using constraints, that are not anti-monotonic can lead to a performance that is far less efficient regarding the pruning process (Boulicaut and Jeudy, 2000) and (Garofalakis et al., 1999), as it would result in an optimization problem and would limit the capabilities of inductive querying.

According to Maimon and Rokach (2010), three algorithms have been proposed to tackle this problem by Srikant et al. (1997). The first two are MultipleJoins and the simplification Reorder, which use constraint relaxation. The third one is Direct, which extends the candidate generation phase. Other approaches mentioned by (Maimon and Rokach, 2010), are the CAP algorithm, capable of using succinct constraints and combinations of constraints by Ng et al. (1998) as well as SPIRIT by Garofalakis et al. (1999), which uses selective relaxation.

Regarding the more complex problem of using anti-monotonic and monotonic constraints in conjunction, Boulicaut and Jeudy (2000) present an algorithm that adapts

MultipleJoins and CAP to this kind of problem. Another approach regarding this problem according to Maimon and Rokach (2010), is known as Molfea, which is an extension to the Max-Miner algorithm by Kramer et al. (2001) and has been used for molecular feature mining.

The major challenges in the field of constraint-based data mining are to generalize the existing methodology in order to make it applicable to many different datasets, especially combined with condensed representation concepts. These problems and possible solutions, as well as more information on the general topic can be found in (Boulicaut and Jeudy, 2000), (Bonchi and Lucchese, 2004) and (Garofalakis and Rastogi, 2000).

2.2.4.3 Advantages and Disadvantages

Following the state-of-the-art literature (Maimon and Rokach, 2010), the advantages and disadvantages can be outlined as follows:

Advantages

- Well studied for item sets and sequences
- If done correctly, provides a generalized framework for knowledge discovery

Disadvantages

- Limited to local pattern mining at the moment
- Has problems dealing with complex constraints and their combination
- There are still many open questions and areas of research

2.2.4.4 Existing Software

Due to the theoretical approach of the general field of constraint-based knowledge discovery methods, most of the algorithms presented are ad-hoc implementations for a very specific set of constraints, tailored to perform an isolated operation on limited datasets. At the time of writing this thesis, no general implementations could be found for the methods described in this section, except for Apriori, which is described in 2.2.3.1. The most likely places to find algorithms based on the concepts of constraint-based methods would be the WEKA framework as well as the Orange knowledge discovery framework.

2.3 Others

This section presents three additional methods of knowledge discovery, namely Neural Networks, methods based on evolutionary algorithms and swarm intelligence algorithms. These methods were selected due to the fact that, in the case of neural networks, they are very versatile and able to perform both supervised and unsupervised learning, and in the case of swarm intelligence and evolutionary algorithms, that they provide methods for problems, which are hard to solve otherwise, such as NP-hard problems. Evolutionary algorithms and methods based on swarm intelligence are also relatively new compared to other, traditional methods and count to the absolute favorites of the author.

2.3.1 Evolutionary Methods

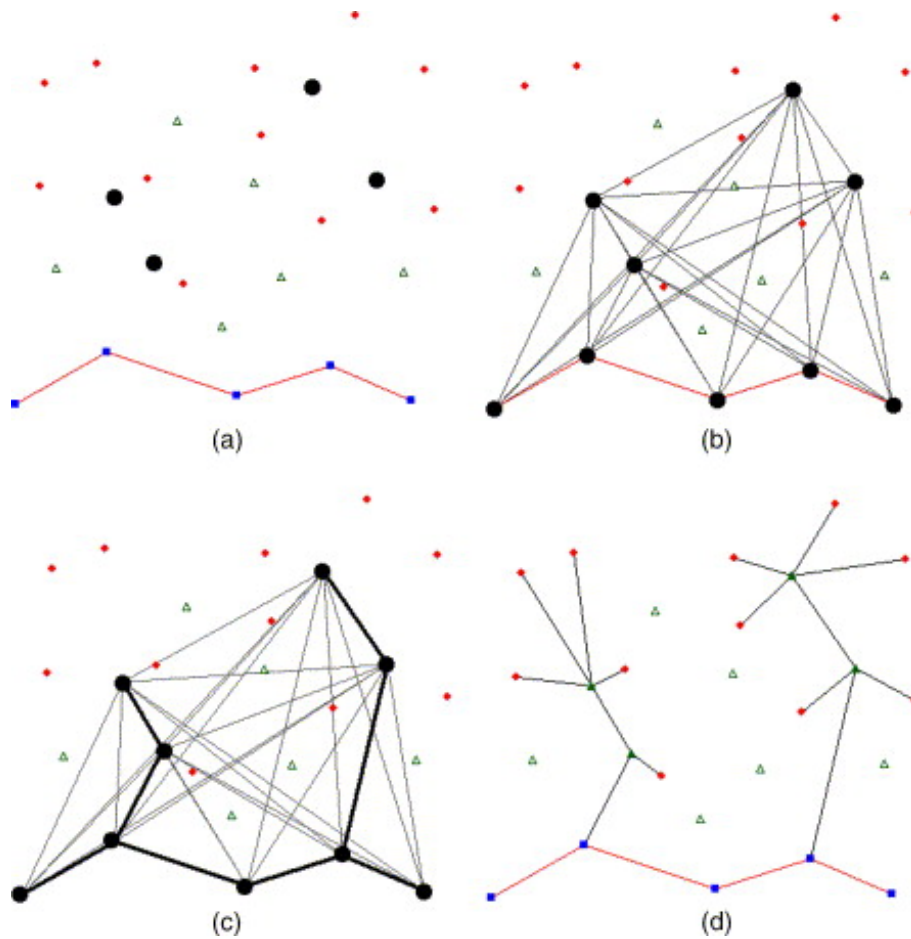


Figure 2.16: Visualization of the solution of a parameter-less evolutionary algorithm for the problem of network expansion by Lobo and Goldberg (2004)

Knowledge discovery methods based on evolutionary algorithms are used mainly for searching and stochastic optimization. They are based on the concepts of natural selection, genetics and the general Darwinian evolution theory (Darwin, 1872). Application areas are obvious fields such as genetics, biology and medicine as well as social sciences, engineering and physics. This survey of available methods will be loosely based on chapter 19 in (Maimon and Rokach, 2010) by Freitas (2002), especially regarding the application of evolutionary algorithms in traditional data mining tasks such as classification and clustering.

The basic idea behind these types of algorithms is, to perform a global search in the search space, assigning them a certain degree of robustness and the ability to

discover data not covered by the methodology that lies behind greedy knowledge discovery methods. In this search space, there are individual candidates, which, at first represent just a random population of solutions for the problem. In steps similar to generations in evolution, these candidates are evaluated using a problem-specific fitness function. Then, the individuals are altered using mutation and crossover, which are both concepts from evolution theory. After their alteration, the newly formed candidates are evaluated anew and then a form of selection takes place, where individuals with low fitness values are replaced by newly formed individuals with higher fitness, thus eliminating weak solutions. This process is repeated until some kind of stopping criterion is reached. EAs are usually not the first choice for problems to which a well established solution method already exists, but even more so for problems where this is not the case, such as with NP-hard problems. The most popular algorithms in the field of methods based on evolutionary algorithms are genetic algorithms and genetic programming, which are described below in more detail.

2.3.1.1 Algorithms

Genetic Algorithms

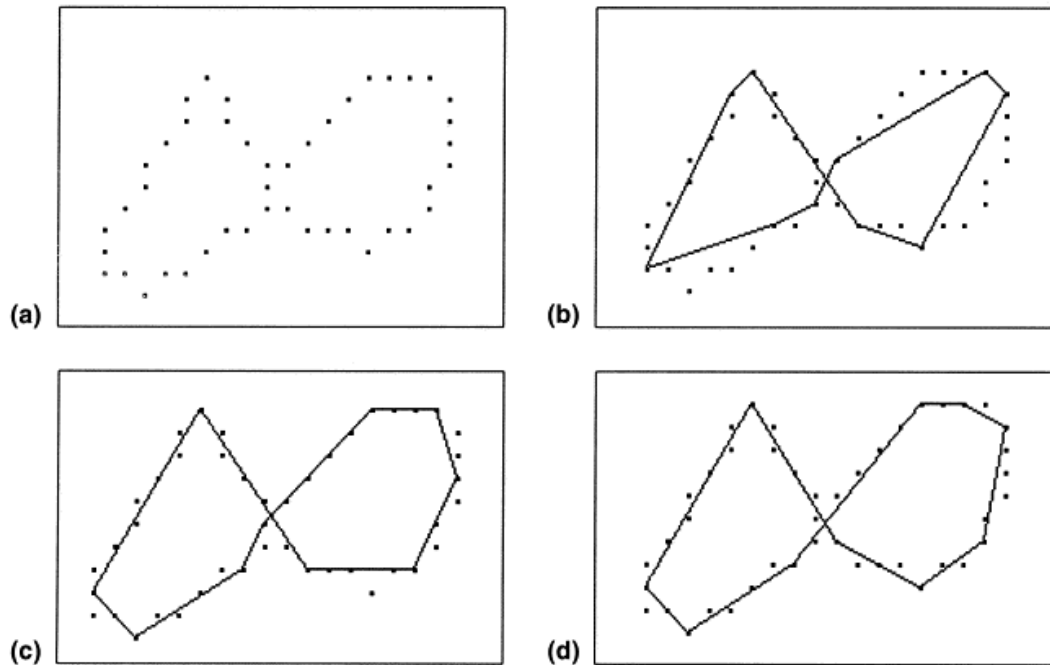


Figure 2.17: Representation of polygonal approximation using genetic algorithms by Yin (1998)

Genetic algorithms are based very closely on the process of natural selection. The search space for such algorithms usually contains vectors of data, which can be of different types, but are optimally numbers of the same length, making the operation of crossover a lot easier. The general process is to initiate the search space with a random population of individual candidates for solutions. Then, the fitness of each individual is evaluated using a fitness function, defined specifically for the problem to be solved. After evaluating each individual's fitness, a loop is started, where the best fits are selected and mutated or crossed. The newly generated individuals are then evaluated based on their fitness and in a selection process, the "weakest" candidates, which are least fit, are replaced by the fittest newly generated candidates. This loop is repeated until a stopping criterion is met, such as a specified number of iterations (generations, steps) or when the fitness of a solution is above a certain threshold. The concept of mutation works by adding, removing or swapping random parts of an individual around. Crossover is a method where candidates are split up at one or

more points and, from the parts of multiple individuals, new candidates are created by merging the parts.

Genetic algorithms usually return individual data points, which might have been altered to fit the problem according to the fitness function.

Genetic Programming

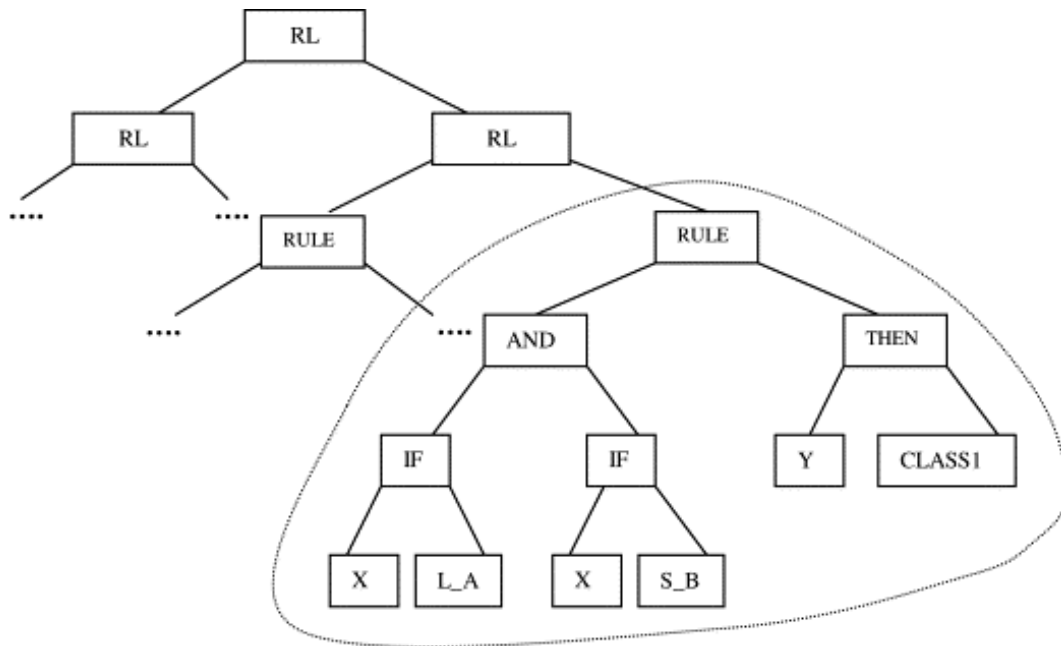


Figure 2.18: Example program in tree form in a genetic programming algorithm (Tsakonas et al., 2004)

The main difference between genetic algorithms and genetic programming are the input and output values, whilst the general process depicting a kind of "natural selection" within data stays the same. Genetic programming methods do not only try to find data, but functions, algorithms and even whole programs, which are able to solve a certain problem. For this purpose, the input data is, for example, a random program or function in the form of a tree, as shown in figure 2.18. Mutations, in this case, describes the process of moving, deleting or swapping nodes of such a tree, in order to generate new individuals, while crossover basically splits up trees and generates new ones from the resulting parts. The concept of evaluation with a fitness function and replacement of weak solutions with fitter ones still applies. Genetic programming algorithms return functions or programs in their intended form, which can then be used to solve a specific problem.

Others

There are also other methods based on evolutionary algorithms, as mentioned in (Maimon and Rokach, 2010), such as Evolution Strategies, Evolution Programming, Learning Classifier Systems (Bull and Kovacs, 2010), Gene expression programming, Neuroevolution and many more as described in (Back, 2000), (Jong, 2006) and (Eiben and Smith, 2003).

2.3.1.2 Classification

Evolutionary algorithms can be used to efficiently perform traditional knowledge discovery tasks such as classification rule mining. (Maimon and Rokach, 2010), provides an overview of approaches and details regarding this topic, which is covered compactly in this section.

A simple approach is to assume that discovered rules have the form

$$IF\{cond\}THEN\{class\}$$

(Witten and Frank, 2005). According to Freitas (2002) and Freitas (2003), there are two approaches regarding the individuals or candidates, namely the Pittsburgh (where one individual contains a whole set of rules) and the Michigan (where each individual contains only one rule) approach. The Pittsburgh method has the advantage, that a whole solution to a classification task is contained in one candidate, meaning that interactions between the rules are already considered (Maimon and Rokach, 2010). The Michigan approach is very simple, but, according to (Freitas, 2002), a good set of rules is preferable to a set of good rules without interactions. (Maimon and Rokach, 2010) mentions some methods to help mitigate this problem by finding more diverse sets of rules using the Michigan approach such as niching (Goldberg and Richardson, 1987) and (Deb and Goldberg, 1989) and sequential covering (Liu and Kwok, 2000), (Zhou et al., 2003) and (Carvalho and Freitas, 2004). Due to the way that rules of classification should interact, they are, according to Maimon and Rokach (2010), presented in the attribute-value form (e.g. "Age < 25" or "Eyecolor = blue") or in the computationally more complex attribute-attribute form (e.g. "Age < Weight" or "NumberOfEyes > NumberOfLegs"). Examples for the usage of attribute-attribute presentations are described in (Hekanaho, 1996) and (Divina and Marchiori, 2002) among others. Generally though, because of the com-

putational complexity and issues regarding the size of the search space, (Maimon and Rokach, 2010) suggests, that attribute-value presentation is being used in most cases. Another approach for rule generation by Jiao et al. (2006) is based on the concept of clustering instances together and finding rules based on these groups. The evaluation of fitness for these classification rules can be performed in various ways, that are often specifically tailored to a particular problem. In general, widely used approaches include the traditional measures of comprehensibility (Fayyad et al., 1996) and surprisingness (Liu et al., 1997), (Romão et al., 2004) and (Freitas, 2006), which are described in detail in (Maimon and Rokach, 2010).

2.3.1.3 Clustering

As described in 2.2.1.4, genetic algorithms can be used for clustering analysis as well. (Maimon and Rokach, 2010) categorizes different clustering approaches using EAs by the following three criteria describing the individual representation based on:

- Cluster description
Individuals represent the parameters to describe the clusters. An individual contains K parameters for K clusters. A whole parameter set describes the properties of the including cluster.
- Centroid / Medoid
Individuals represent the coordinates of a clusters Centroid (center) or Medoid (which is the representative, most central point)
- Instance
Individuals represent strings of n elements, where n is the amount of data instances. The strings are also called genes and specify the index of the cluster in which the instances are contained.

More detailed information on representations as well as a comparison between those mentioned above, can be found in (Maimon and Rokach, 2010). As stated in 2.2.1.4, the evaluation of unsupervised methods, such as clustering analysis, can be very difficult, but there are fitness evaluation methods available for EAs in the context of clustering, which mainly use the traditional measures of similarity and distance

among others and are described in (Kim et al., 2000), (Handl et al., 2004), (Korkmaz et al., 2006), and (Hall et al., 1999).

2.3.1.4 Advantages and Disadvantages

Considering some of the state-of-the-art literature (Maimon and Rokach, 2010), the advantages and disadvantages can be summed up as follows:

Advantages

- Well suited for parallel computing
- Robust to noise
- Good choice for problems, where no traditional method already provides a solid foundation
- Scales well to high dimensional problems
- Easy to implement and adjust
- Good for multiple-objective problems

Disadvantages

- Does not guarantee a global maximum, thus no optimal solution, in finite time
- Not the first choice for problems for which there are well studied methods
- Computationally expensive

2.3.1.5 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- KEEL - <http://www.keel.es/>
- SolveIT - <http://www.solveitsoftware.com/>
- MCMLL - <http://mcml.sourceforge.net/>
- JGAP - <http://jgap.sourceforge.net/>
- SAS - <https://www.sas.com/>

2.3.2 Neural Networks

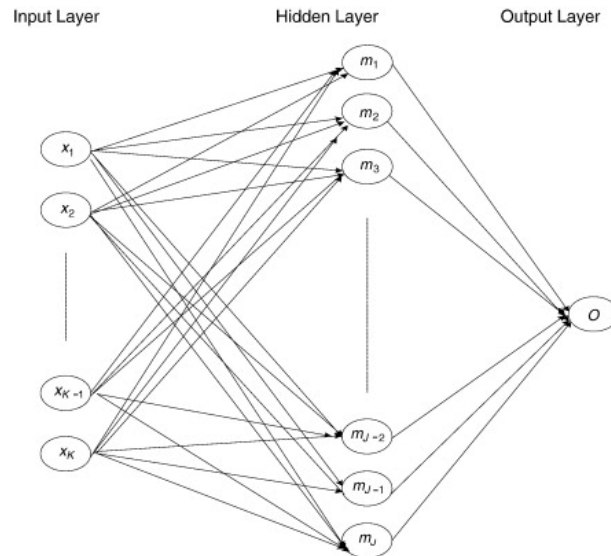


Figure 2.19: Visualization of a single layer feed forward neural network (Branch et al., 2008)

Neural Networks count to the best known knowledge discovery methods due to them having been existent since the dawn of the computerized age. Their applications lie mainly in fields where comprehensibility is not so important and not a lot of explicit knowledge about the problem exists. Examples for areas, where neural networks have had a great impact are pattern recognition (e.g. signals or images), sound synthesizing and robotics. Neural networks are well suited for nonlinear, highly complex problems, which would be hard to model with other methods and are fairly robust to noise and missing data according to Maimon and Rokach (2010). These networks are also able to perform data mining tasks like classification (Dutta and Shenkar, 1993), (He et al., 1997) and (Zhang et al., 2003), association (Paik and Katsaggelos, 1992), (Tatem et al., 2002), and (Changchien and Lu, 2001), as well as cluster analysis (Dittenbach et al., 2002), (Smith and Ng, 2003) and (Lewis et al., 1997). There are, however, many further examples of using neural networks within the area of data mining. A detailed list can be found in (Maimon and Rokach, 2010), which this section of the survey will be based on, regarding the methods and references covered.

The basic idea behind methods based on artificial neural networks, which have their roots in artificial intelligence, is to simulate the interconnectivity in the human

brain. According to Maimon and Rokach (2010), the factors, that are attempted to be simulated are the ability to process information in a parallel way on the one hand, and to learn from experience on the other. The three most commonly used models within the field of neural networks, mentioned in (Maimon and Rokach, 2010), namely Feedforward Multiayer Networks (MLP), Hopfield Networks and Kohonen's Self organized maps, are described in the following sections. For more information on the vast history of neural networks and further general information, the reader can consult (Hastie et al., 2008) and (Maimon and Rokach, 2010).

2.3.2.1 Models

Feedforward neural networks

MLP - multi layer perceptrons or feedforward neural networks are the most popular method in the field and are mainly used to model relationships between predictor- and response variables. The method is well suited for forecast and classification purposes and is based on, or highly related to statistical pattern recognition methods (Jensen and Kendall, 1993) and (Cheng and Titterington, 1994).

Generally, neural networks consist of interconnected nodes (neurons). Each node receives inputs, calculates them based on a certain transfer function and generates an output. The edges of a neural network have a weight value, describing the strength of the relationship between neurons. In the specific case of feedforward neural networks, there are at least 3 layers, an input layer, a hidden layer and an output layer as visualized in figure 2.19. There may be more than one hidden layer, depending on the application. The input layer represents the predictor variables while the output layer represents the response variables. The hidden layer, however, which is connected to both input and output neurons, uses complex, problem based transfer functions to generate information based on the input and sends them to the output neurons. In MLPs, there is only one direction: input - hidden - output, meaning that no feedback mechanisms have been established.

According to Maimon and Rokach (2010), neurons process information in two steps. In the first step, all of the connected inputs are summed up, forming a weighted sum of input values and their weights (i.e. the links connecting them). The second step uses the neuron's transfer function to convert this weighted sum to an output, using (Maimon and Rokach, 2010):

$$Out_n = f\left(\sum_i w_i x_i\right). \quad (2.21)$$

Where x_i are the inputs, f is the transfer function and w_i are their weights. In (Maimon and Rokach, 2010), some of the most commonly used transfer functions are listed, with the logistic function mentioned as the most popular one.

The training of neural networks resembling the form of MLPs works in the traditional supervised learning way, where inputs and known outputs are used and methods such as MSE or SSE are used to evaluate the training and test performance of the generated networks. Another approach mentioned in (Maimon and Rokach, 2010) is that of using backpropagation with gradient steepest descent for training.

In depth information on factors to consider when creating a neural network model, as well as potential pitfalls and best practices can be found in the highly elaborate and well written description of the process in (Maimon and Rokach, 2010).

Hopfield networks

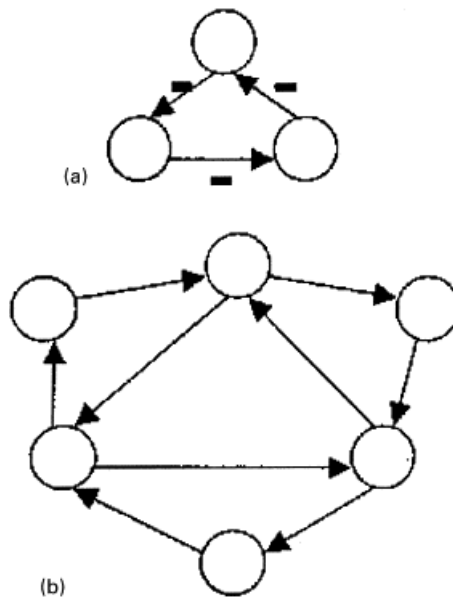


Figure 2.20: A three unit Hopfield Network (a) and adding three units to it (b) (Bersini and Sener, 2002)

Hopfield networks by Hopfield (1982), are very different from the traditional neural network approach used in multi layer perceptrons. The most obvious difference is, that there is no layer structure in Hopfield networks, but a single layer containing all neurons. The neurons on that layer are completely interconnected and, contrary to feedforward approaches, can contain loops. An example of a Hopfield network is visualized in figure 2.20. Every neuron in a Hopfield network is both an input- and output neuron at the same time and can be described as (Maimon and Rokach, 2010):

$$u_i(t) = \sum_{j=1}^n w_{ij}x_j(t) + v_i \quad (2.22)$$

Where $x_i(t)$ is the neuron's output state, v_i the neuron's threshold and $u_i(t)$ is the neuron's internal state.

According to Maimon and Rokach (2010), Hopfield networks are mostly used as associative memories, using patterns as inputs, which are noisy, blurred or altered otherwise and then finding the original, clear pattern within their memory as a result. Associative memory means, that patterns are stored within the network in the form of weights, which can be defined as (Maimon and Rokach, 2010):

$$w = \frac{1}{n} \sum_{i=1}^p z_i z_i^T \quad (2.23)$$

Where z_i are patterns. This works very well for a limited amount of input patterns, but has the drawback that, if there are many very similar patterns or just a very high amount of different ones, the network becomes unstable.

Self organized maps

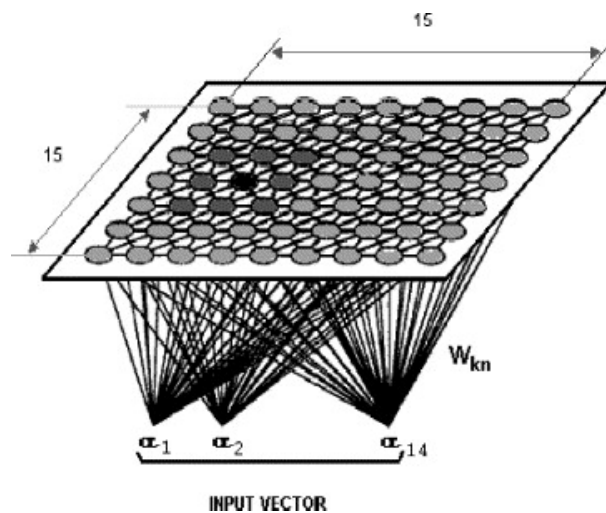


Figure 2.21: Visualization of a 15x15 Kohonen Net (Wyns et al., 2004)

SOM - Self Organizing Maps by Kohonen (1990) are a method to perform mainly clustering and dimensional reduction tasks. The basic idea is to create a lower dimension map of a high dimensional input. This method is, again, very different from the traditional neural network approach in MLPs, because there are no known outputs, which makes supervised learning impossible (Maimon and Rokach, 2010). The clustering of data works by adjusting weights inside of a so called Kohonen Map as shown in figure 2.21, where there are two layers of nodes - input and output - that are fully connected to each other. The number of nodes to be formed in order to

create enough clusters has been recommended to be about ten times the dimension of the input by Deboeck and Kohonen (1998).

In the training process a Self Organizing Map, the neurons evaluate the similarity between their weights and the input. The neuron, whose weights are the most similar to the pattern is the winner (Maimon and Rokach, 2010), which results in an increase of the neuron's and it's neighborhood's weights in regard to the input pattern. This method is called unsupervised and competitive with winner-take-all strategy according to Maimon and Rokach (2010), which also provides a more detailed step-by-step description of the training process.

After the training of the self organized map is completed, the model can be used to visualize the structure of the clusters and their properties.

2.3.2.2 Advantages and Disadvantages

Considering some state-of-the-art literature (Maimon and Rokach, 2010), the advantages and disadvantages can be listed as follows:

Advantages

- Easy to implement
- Robust to noise and missing data
- Well suited for high dimension problems
- Good at solving problems without having a lot of context knowledge about it

Disadvantages

- Computationally slow
- Not comprehensible
- Tends to overfitting
- Cannot guarantee if solution is a global or local maximum

2.3.2.3 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- Orange - <http://orange.biolab.si/>
- MemBrain - <http://www.membrain-nn.de/>
- SAS - <https://www.sas.com/>
- KNIME - <http://www.knime.org/>
- MATLAB - <http://www.mathworks.com/>
- Mathematica - <http://www.wolfram.com/mathematica/>
- SPSS Neural Networks - <http://www-01.ibm.com/software/analytics/spss/>

2.3.3 Swarm Intelligence Methods

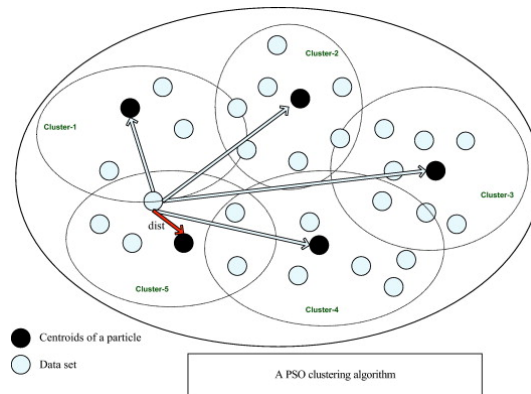


Figure 2.22: Example visualization of Particle Swarm Optimization in the context of clustering (Chang et al., 2012)

Knowledge discovery methods based on the concepts of swarm intelligence (Beni and Wang, 1989) are agent based algorithms, which use the behavior often observed in groups of animals such as fish or ants, where individuals of a group are only able to perform simple actions, whereas in a global context, collective behavior comprising the information of all individuals can be used to achieve complex goals. The field of swarm intelligence in data mining applications is relatively young, yet there are a number of areas where techniques of this type have been applied successfully such as robotics, medicine, geography, social sciences and especially route optimization. This short overview over some of the methods and applications in this field is based on chapter 23 in (Maimon and Rokach, 2010) and focuses on the two most commonly used techniques, ACO - Ant Colony Optimization (Dorigo et al., 1996) and PSO - Particle Swarm Optimization (Kennedy and Eberhart, 1995). (Millonas, 1992) suggests, that there are five essential principles to define swarm intelligence: Proximity, Quality, Diverse Response Ability, Stability and Adaptability. A swarm with these properties is able to move and react uniformly and quickly to dangers as well as opportunities, while still staying relatively adaptable to change, if necessary. Other relevant and interesting concepts regarding collective swarm behavior in both a global and a dynamic context, can be found in (Couzin et al., 2002). Methods based on swarm intelligence are very well suited for solving NP-hard problems such as the traveling salesman problem as well as other problems, where there are no established, well working methods available.

2.3.3.1 ACO

Ant Colony Optimization, based on Ant Colony Systems by Dorigo and Gambardella (1997), is a knowledge discovery method that considers the natural behavior of ants searching for food. The general idea, is that ants, when searching for food, prefer to walk on paths with the maximum amount of pheromones available, which are the ones that a lot of ants have used to successfully find food. Based on this relatively simple concept, according to Maimon and Rokach (2010), the goal of ACO is to find a path with minimal cost in a graph. The motivation behind ACO was to efficiently solve the NP-hard traveling salesman problem, which is a combinatorial optimization problem, where the goal is to find the shortest path in a graph, visiting each node exactly once and then returning to the origin. ACO basically works by letting virtual ants walk down the different paths in the graph, leaving behind pheromones - in this case just a weighting value - on the edges. These pheromones have a decay function, thus reducing them over time. By individual ants working together, sharing information between each other, different paths are tested and evaluated, where bad paths lose their pheromone level over time and good paths increase theirs, leading to an optimal way of traversing the nodes in the end.

Based on (Maimon and Rokach, 2010), this process can be formulated as follows. There are two updating functions for pheromones, one local function for ants visiting edges and one global function, modifying the pheromone level on the best traversal path found so far. Each function has a decay parameter built-in, reducing the overall pheromone level over time at all edges. After the initialization, for each global iteration, all ants are assigned a starting node. Then, for each step inside these iterations, the ants build a solution and update pheromones locally, until all ants have performed a complete traversal. At the end of each inner step, the global pheromone update is performed, providing the ants with globally available information about the best paths so far. The iterations are repeated until a stopping criterion is reached. Detailed formulas for the different functions and a comprehensible pseudo code for the ACO procedure can be found in (Maimon and Rokach, 2010). Other variations of the ant colony system are the Elitist Ant System (Negulescu et al., 2008), Max-Min Ant System (Stützle and Hoos, 2000) (Bullnheimer et al., 1997) and the Continuous Orthogonal Ant Colony (COAC) (Hu et al., 2008).

2.3.3.2 PSO

PSO - Particle Swarm Optimization was originally developed for simulating social behavior by Kennedy and Eberhart (1995). It is a method based on metaheuristics and is mainly used for search and optimization problems. The concept behind it is rather simple, there are a set of particles and each of these particles has coordinates describing its position and a velocity. Then, for each particle, a function is evaluated using the particle's position. (Maimon and Rokach, 2010) provides a detailed explanation of possible generic update functions for both the position and the velocity of a particle. The algorithm searches for the best global solution g while each particle has a best local solution P . After the initialization of a random population of particles in the search space, with random values for both position and velocity for each particle, for every iteration, for each particle the particle's fitness according to the problem is calculated using the function f . After evaluating the particle's fitness, if it is a new local best solution, P is updated and if it is also a better fit than g , g is updated. At the end of the step, the particle's position is updated based on an update formula. This process is repeated until a stopping criterion such as a maximum number of iterations of the solution's fitness being above a threshold, is reached.

2.3.3.3 Advantages and Disadvantages

Following some of the state-of-the-art literature (Maimon and Rokach, 2010), the advantages and disadvantages can be summarized as follows:

Advantages

- Easy to implement
- provides dynamic properties for adaptation
- Good option for NP-hard problems like TSP
- Well suited for parallel computation

Disadvantages

- More experimental than theoretical approaches
- Limited comprehensibility
- Cannot guarantee if solution is a global or local maximum

2.3.3.4 Existing Software

- WEKA - <http://www.cs.waikato.ac.nz/ml/weka/>
- antcolony - <http://sourceforge.net/projects/antcolony/>
- ant colony optimization - <http://www.aco-metaheuristic.org/>
- MIDACO - <http://www.midaco-solver.com/>
- several applets & programs - <http://www.particleswarm.info/>
- MATLAB - <http://www.mathworks.com/>

3. Related Work

3.1 Literature

This section presents a short overview over some of the basic literature in the field of knowledge discovery, which was used to write this thesis. Of course this is just a small part of the available literature on this vast topic, but for readers who are interested in getting a better general understanding or just a few different views on the subject of knowledge discovery and the discussed methods, the following list provides some good pointers on where to look.

3.1.1 Books

Data Mining and Knowledge Discovery Handbook, (Maimon and Rokach, 2010) This extensive work by Oded Maimon and Lior Rokach covers just about all relevant topics related to data mining and knowledge discovery, making it an amazing work of reference. The book consists of eight chapters, seven of them regarding methods used in KDD/DM and the eighth providing a survey of software and tools available for these methods. All methods are described in a manner focused on theoretical details regarding the algorithms behind them and with a critical view on their advantages, disadvantages and practical applications.

The Elements of Statistical Learning, (Hastie et al., 2008) This book by Trevor Hastie, Robert Tibshirani and Jerome Friedman is another fairly extensive collection of knowledge discovery methods. The book describes the concepts behind the methods rather than mathematical details and provides the reader with techniques to judge and evaluate them in regard to their usefulness and performance in different fields. It is another very good work of reference in the field, which gives an insight into the technical concept behind machine learning using intuitive examples.

The Top Ten Algorithms in Data Mining, (Wu and Kumar, 2009) This relatively short book by Xindong Wu and Vipin Kumar is an attempt to describe the top 10 algorithms in data mining identified in 2006 by the IEEE International Conference on Data Mining. It includes insights into the technical details of C4.5, k-Means, SVM, Apriori, EM, PageRank, AdaBoost, k-Nearest-Neighbor, Naïve Bayes and CART. Each chapter also provides some exercises, a summary and, in most cases, a short survey of existing software implementations of the method in question.

Cluster Analysis, (Everitt et al., 2009) This book by Brian Everitt, Sabine Landau and Morven Leese describes one of the most popular categories of knowledge discovery methods: clustering analysis methods. It explains concepts, both old and new, concerning different types of clustering methods. The book also provides some insight into the areas of application of cluster analysis as well as possibilities to compare and evaluate clustering methods.

3.1.2 Journal Papers

There are, of course, a lot of papers dealing with problems and methods of knowledge discovery. In the following paragraph, a few general papers as well as some of the works describing the basic methods mentioned in the previous chapter are named to provide the reader with a few directions in regard to acquiring more information on the subject of knowledge discovery.

The KDD Process for Extracting Useful Knowledge from Volumes of Data, (Fayyad et al., 1996), by Usama Fayyad, Gergory Piatetsky-Shapiro and Padhraic Smyth describes the knowledge discovery process and the steps necessary to use it successfully. Supervised Machine Learning: A Review of Classification Techniques, (Kotsiantis, 2007) by Sotiris B. Kotsiantis does the same with a focus on the supervised machine learning process.

One of the most defining works in the area of classification trees was Classification and Regression Trees (Breiman et al., 1984) by Leo Breiman, Jerome H. Friedman, Richard A. Olshen and Charles J. Stone, presenting the popular CART approach, which is one of the most widely used data mining techniques nowadays.

An extensive work, especially in regard to support vector machines, was published in 1995 by Vladimir N. Vapnik with The nature of statistical learning theory (Vapnik, 1995).

Eugene Charniak, in his 1991 article called Bayesian Networks Without Tears (Charniak, 1991), provides a nice and comprehensible introduction to Bayesian Networks for researchers without extensive experience or knowledge regarding probability theory.

MARS, another widely used technique, was presented in Multivariate Adaptive Regression Splines (Friedman, 1991), by Jerome H. Friedman, who was one of the founders of CART.

Another highly popular algorithm, called Apriori, was developed and introduced right after publishing Fast Algorithms for Mining Association Rules (Agrawal and Srikant, 1994) by Rakesh Agrawal and Ramakrishnan Srikant, which is one of the groundbreaking works in the field of association rule mining.

Alex A. Freitas, an authority in the field of data mining, especially based on evolutionary algorithms, published A survey of evolutionary algorithms for data mining

and knowledge discovery (Freitas, 2003), which is a very good and comprehensible overview of evolutionary algorithms and their applications within knowledge discovery.

Last, but definitely not least in this short list of literature is Swarm intelligence in cellular robotic systems (Beni and Wang, 1989), by Gerardo Beni and Jing Wang, which is one of the first works in the area of using swarm intelligence in the context of knowledge discovery and information technology in general.

3.2 Software

This section provides a description and short survey of a few software implementations for knowledge discovery methods with a focus on open source frameworks. Obviously, the following list only presents a small sample of the available software solutions regarding knowledge discovery and data mining. The applications listed here were chosen for their popularity in literature as well as for criteria such as usability, money implications, extensibility and how they relate to the problem of bringing knowledge discovery methods to more people in different disciplines of science.

WEKA, <http://www.cs.waikato.ac.nz/ml/weka/>

The WEKA Data Mining Software is a collection of machine learning algorithms based on the Java programming language. It is an open source framework issued under the GNU General Public License and provides both the possibility to be used as a library for embedding and as a standalone GUI-software. WEKA implements a wide variety of algorithms and visualizations, which are regularly extended by a very active community. The framework is well documented, but the user interface is not suited for unexperienced users without a certain knowledge of the underlying methods. The methods however are well implemented and the framework provides a high functionality for experienced users. More information on WEKA can be found in (Holmes et al., 1994).

Advantages

- Open Source
- Very extensive collection of algorithms
- Well documented

Disadvantages

- GUI only partly available
- Not easily extendable
- Need to download software

Orange, <http://orange.biolab.si/>

Orange is a data mining, visualization and analysis tool suitable for both unexperienced users and experts. The Application can be extended fairly easily via Python scripting, which includes algorithms as well as visualization methods. The user interface of most of the methods is fairly intuitive, using data-pipelines and drag-and-droppable icons to create models with, in some cases, a limited amount of required knowledge about the underlying algorithm. The application is open source under the terms of the GNU General Public License. More information on Orange can be found in (Demšar et al., 2004).

Advantages

- Open Source
- Many implementations
- Fairly easy to extend (Python scripting)
- Very Modular
- Well implemented visualizations and GUI's

Disadvantages

- Need to download software
- Long installation procedure (Python)

KNIME

KNIME, the Konstanz Information Miner, developed at the University of Konstanz, Germany, is an open-source data integration, analysis and exploration platform licensed under the GNU General Public License. The application is based on the Eclipse (<http://www.eclipse.org/>) platform, making it possible to extend it to some degree. KNIME is highly modular and the user interface is very intuitive, especially for people who have worked with the Eclipse IDE before, also using drag-and-drop and data-pipelines to construct comprehensible models without the need to know every detail of the underlying algorithms. More information on the KNIME platform can be found in (Berthold et al., 2008).

Advantages

- Open Source
- Modular approach
- intuitive GUI
- Many algorithms

Disadvantages

- Not easy to extend
- Need to download

Salford Systems (CART), <http://www.salford-systems.com>

The American company Salford Systems has several award-winning products to offer for knowledge discovery and data mining, such as MARS, LOGIT, TreeNet, SPM and most prominently, CART. The CART implementation of Salford Systems is the only implementation using the original proprietary code by Breiman et al. (1984). Although the tools are commercially distributed, there are researcher licensing options available for students. Because of the long term success of CART and the other products, they are very refined regarding both the performance and the usability. Clear disadvantages, beside the non-open-source availability are, that every tool only contains one, although very well implemented method and that the tools are mostly focused on business problems. More information on CART by Salford Systems can be found in (Wass, 2007).

Advantages

- Licensing for Researchers and Students available
- Award-winning proprietary algorithms
- Very good Implementations
- Many features for Visualization
- Good usability

Disadvantages

- Commercial Software
- Only one Algorithm per Application

SAS Enterprise Miner, <http://www.sas.com>

The SAS Enterprise Miner, created by the American company SAS, is considered one of the most popular commercially available data mining software. There are a lot of modules available to extend the basic functionality of the application and to customize it for certain projects. The Enterprise miner implements a wide variety of data mining and knowledge discovery methods as well as powerful visualization tools, while still being fairly easy to handle, even for less experienced users. More information on the SAS Enterprise Miner can be found in (Cerrito, 2006).

Advantages

- Good usability
- Many algorithms
- Customizable
- Powerful Visualization

Disadvantages

- Commercial Software
- Modular approach, but modules have to be acquired one by one

Others

Further information on the subject as well as reviews and evaluations of the applications mentioned above and others can be found in (Maimon and Rokach, 2010), (Wu and Kumar, 2009), (Haughton et al., 2003), (Zhang, 2010) and (Chen et al., 2007).

This page intentionally left blank

4. Materials and Methods

Home Upload Create Manage Help Logout

Hello, admin

admin Logout
ACTIONS
Settings
Administration

Projects leading:

Filter

- [ClassificationDemo](#)

Subprojects as Researcher:

Filter

- [CART](#)
- [C45](#)

Results

Filter

creation Date	Method	Creator	Details
09/02/2012 18:00	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:57	C45	admin	Details

Figure 4.1: Screenshot of the "Manage Projects"-Area of the KNODWAT application.

This chapter provides a detailed overview of the software engineering process behind the creation of KNODWAT - Knowledge Discovery With Advanced Techniques, a highly extensible framework application for testing knowledge discovery techniques. The first section describes some of the existing technology used for developing the application and provides some reasoning in regard to why these technologies were chosen over others. This part is followed by a detailed description of the application and its structure, giving an overview of the whole system at first and then going into technical details of the application's core features.

4.1 Technology

4.1.1 Spring

<http://www.springsource.org/>

The Spring Framework (used version: 3.1.0), is an open source application framework under the Apache 2.0 license, written in the Java programming language. It is widely used and quite popular, especially for Enterprise Java Web applications. Spring includes a wide range of modules, providing for example AOP - Aspect Oriented Programming (Holzinger et al., 2011) functionality, a MVC - Model View Controller (Holzinger et al., 2010) framework as well as several modules for data access, performance enhancement and security. There are several reasons, why this framework has been chosen as a basis for KNODWAT, such as the framework documentation provided as well as the user generated documentation on the web, as a result of the popularity of the framework. Other reasons are multi platform support and the integrated support for other popular open source frameworks such as Hibernate. The most important factor, however, was personal experience, as the developer of KNODWAT has worked with Spring extensively on multiple projects before. Spring provides a very solid basis, especially in the context of an MVC-based, extensible application framework using core concepts such as Inversion of Control and Convention over Configuration.

4.1.1.1 Spring Security

<http://www.springsource.org/spring-security>

As its name suggests, Spring Security (used version: 3.1.0) is an access-control and authentication framework targeted at Spring applications. Due to its close interaction and support within the Spring framework, Spring Security was the obvious choice for the login-functionality as well as for the restriction of certain areas for different user roles.

4.1.2 Hibernate

<http://www.hibernate.org/>

The Hibernate framework (used version: 3.5.1) is basically a library used for mapping Java objects to a relational database. The framework also has very good Spring integration and supports multiple database types such as MySQL, Oracle, HSQL and many others, making it a very good choice in regard to multiplatform compatibility. Hibernate also provides HQL, the Hibernate Query Language, which is an SQL-like language, making it possible to use the same queries on different databases. It is licensed under the GNU Lesser General Public License.

4.1.3 jQuery

<http://jquery.com/>

The jQuery open source JavaScript library (used version: 1.7.2), licensed under the MIT license, is one of the most widely used piece of technology in modern websites. Considering the vast amounts of documentation and examples, the multi-browser support and the large quantity of already existing widgets it provides, it is an obvious choice. jQuery makes it easy to handle events, as well as to animate and manipulate parts of HTML, making it a very important tool for improving the usability of a web application.

4.1.3.1 Blueimp

<https://github.com/blueimp/jQuery-File-Upload>

The Blueimp jQuery File Upload Widget is an open source frontend for File Upload based on the jQuery JavaScript library and Bootstrap. The developer chose this widget, because it looked very good out of the box and provided important usability features such as progress bars, multiple file upload and file preview, as well as an easy way to customize almost everything.

4.1.3.2 Chosen

<http://harvesthq.github.com/chosen/>

Chosen is another jQuery plugin, which improves the usability of select- and multi-select fields, by extending them with a text filter and improving their general look and feel. This plugin was chosen due to the relatively high amount of select fields inside the KNODWAT application, making it easier for inexperienced users to create and edit content.

4.1.4 Bootstrap

<http://twitter.github.com/bootstrap/>

The Bootstrap toolkit (used version: 2.1.1) by Twitter, provides a well documented, easy to use toolbox for CSS, JavaScript and HTML styling. It is widely used within jQuery-based widgets and greatly reduces the effort necessary to create a homogeneous, attractive design. Bootstrap is licensed under the Apache 2.0 license. The developer's limited experience in designing complex markup with CSS and HTML, as well as his existing experience in using Bootstrap were the main reasons for including it in the KNODWAT project.

4.1.5 Others

4.1.5.1 WEKA

<http://www.cs.waikato.ac.nz/ml/weka/>

The WEKA Data Mining Software is described in 3.2. WEKA is GPL licensed and provides not only a standalone application, but also a Java library for using the implemented algorithms in Java applications. In KNODWAT, WEKA 3.7.7 was used to implement the two example methods, C4.5 and CART. The WEKA library is well documented and includes examples and sample data, which made it an easy choice to use it for KNODWAT.

4.1.5.2 JIT

<http://thejit.org/>

The JIT - JavaScript InfoVis Toolkit (used version: 2.0.1) is a BSD-licensed open source library for visualizing data sets on the web. The library is well documented and provides examples and sample data for visualizing graphs such as trees and pie charts. JIT was used to visualize the amount of correctly and incorrectly classified instances of a data set in KNODWAT, due to the fact that it is powerful, good looking and easy to use.

4.1.5.3 JACKSON JSON

<http://jackson.codehaus.org/>

The JACKSON JSON processor library (used version: 1.9.6), licensed under the GPL, has been used to create the JSON data that is used as a source for the JIT visualization library. It is easy to use and well documented, but there are many other similarly good options available in Java.

4.1.6 Development Tools

The following list shows the tools that were used during the development of KNODWAT. They were all chosen due to personal preference and experience of the developer.

- Netbeans 7.1.1, <http://netbeans.org/>
- Tomcat 7.0, <http://tomcat.apache.org/>
- MySQL Server 5.5, <http://www.mysql.com/>
- Maven 2.3.1, <http://maven.apache.org/>
- spring archetypes, <http://code.google.com/p/spring-archetypes/>

4.2 Application Structure

This section both provides an overview of the software architecture behind the KNODWAT application, as well as profound information on technical details, design decisions and screenshots showing the features of the software. In general, the application was implemented using the MVC - Model View Controller design pattern (Holzinger et al., 2010), which means that the software is structured in three separate units. In this case, the model represents the data which has to be shown to the user, such as projects and results. The view is responsible for the presentation of the data and for handling user interaction, and in the case of KNODWAT, this unit is the web-based graphical user interface. The controller can send commands to both the view and the model, manipulating the data and the presentation layer. This approach enables the reuse and extension of each separate unit, making the software very flexible. The Spring framework includes sophisticated MVC-functionality, making it fairly simple to construct applications based on this design principle, which was another reason to use it.

Probably the most important architectural, or rather general design decision regarding the KNODWAT framework was to make it a web based application. This crucial decision was based on three factors: usability, multi platform compatibility and the social aspect of research. With KNODWAT being aimed at researchers from all disciplines of science, especially at people with little experience in information technology, introducing new users to the framework will be easier if the user interface is similar to widely used services such as Twitter and Youtube. Even with little IT knowledge and experience, there is a high chance, that a researcher has been using the web including search engines and social networks to communicate and find resources. This assumption leads to the conclusion, that a web interface based on the general design principles of well known services, should make it easier for inexperienced users to be introduced to the application. The second concern was multi platform availability for the application. Due to the popularity of both smartphones and tablets, there is an inherent need to make applications available to static and mobile devices. This can be difficult, considering the different technologies involved in creating mobile applications (Android, iOS, Windows Mobile..), but all of these devices have web browsers installed, capable of displaying complex web applications

and enabling user interaction on the same level as a PC can. The third reason for making KNODWAT web-based was the social component of research, meaning the creation and sharing of results with other people, or following other researcher's progress.

4.2.1 Data model

The data model behind the KNODWAT framework was built using object relational mapping, which means that, using the Hibernate library, the tables were initially created as Java objects. These Java classes include properties such as a unique identifier and other context specific properties like a name or creation date. They also provide getter- and setter methods for these properties. The class itself is annotated with the `@Entity` annotation, telling the Hibernate framework, that this class is a database entity (table). Primary keys are annotated with `@Id`. There are other annotations as well, describing how relations should be constructed within the database such as `@ManyToOne`, or how a certain data type should be created (e.g. a date with `@Temporal`), telling the Hibernate framework to save this date as a UNIX timestamp. An example of such an annotated entity class is shown in figure 4.2.

```
@Entity
@Table(name="comment")
public class Comment implements Serializable {

    @Id
    @GeneratedValue
    private int id;

    @ManyToOne
    @JoinColumn(name="idCreator")
    private User creator;
    @Column(columnDefinition="text")
    @NotNull
    private String text;
    @Temporal(TemporalType.TIMESTAMP)
    private Date createDate;
    @ManyToOne
    @JoinColumn(name="idProject")
    private Project pproject;
    @ManyToOne
    @JoinColumn(name="idSubProject")
    private SubProject subProject;
    @ManyToOne
    @JoinColumn(name="idResult")
    private Result result;

    public Result getResult() {
        return result;
    }

    public void setResult(Result result) {
        this.result = result;
    }

    public SubProject getSubProject() {
        return subProject;
    }
}
```

I

Figure 4.2: The entity class for the comments-table, showing the annotations used for the object relational mapping.

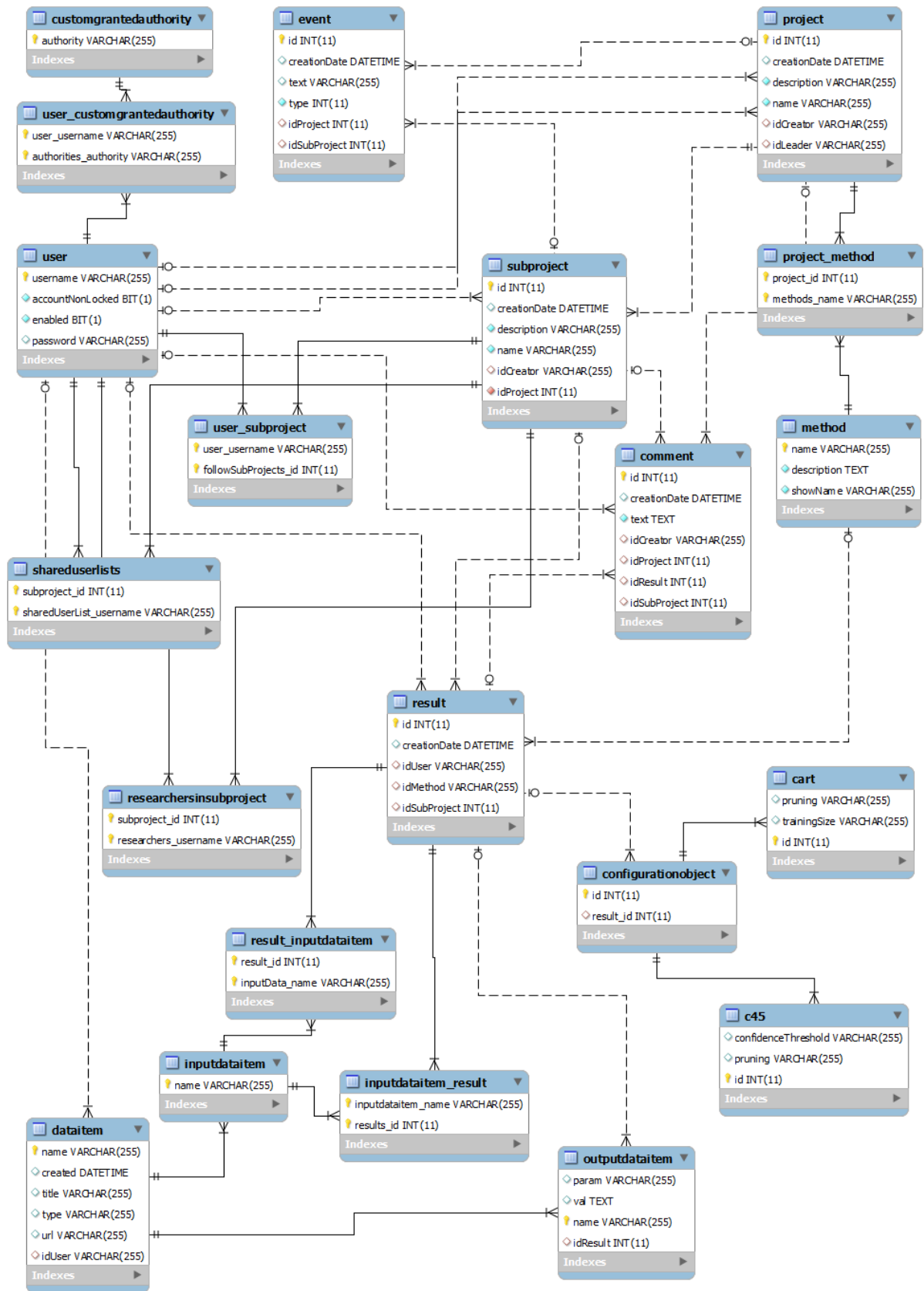


Figure 4.3: Entity-relationship diagram, generated from the live database of the KNODWAT application.

Using these model-classes and some kind of configuration with respect to the database type and server to be used, the Hibernate framework creates a complete data model, as shown in figure 4.3, and can then be accessed from within the Java application by using the entity classes. This data model was created from the object relational mapping of the entity classes presented in figure 4.4. One of the most central pieces of this data model is the User object, which is connected to almost all the other entities by either being a researcher of a project, a creator of another object or a project leader. This can be easily explained by the KNODWAT platform being based on the Web 2.0 concept of user generated content. Another interesting interaction is the one between a project, a subproject and a result. A project is basically an organizational unit with a leader, who can administrate subprojects. These subprojects are the actual research projects, where results are generated and shared between users by researchers, who can be added or removed by the project leader of the subproject's parent project. Projects include a set of different methods, which basically are the knowledge discovery algorithms implemented in the platform. These methods can be used subsequently to create results within subprojects. Results are created by using input data and configuration objects that, describing the parameters used for the chosen algorithm. Once the result creation has been performed, output data is created and saved in a Result object. Users can comment on results, subprojects and projects and are notified via events, if something new has occurred in a subproject they are following.

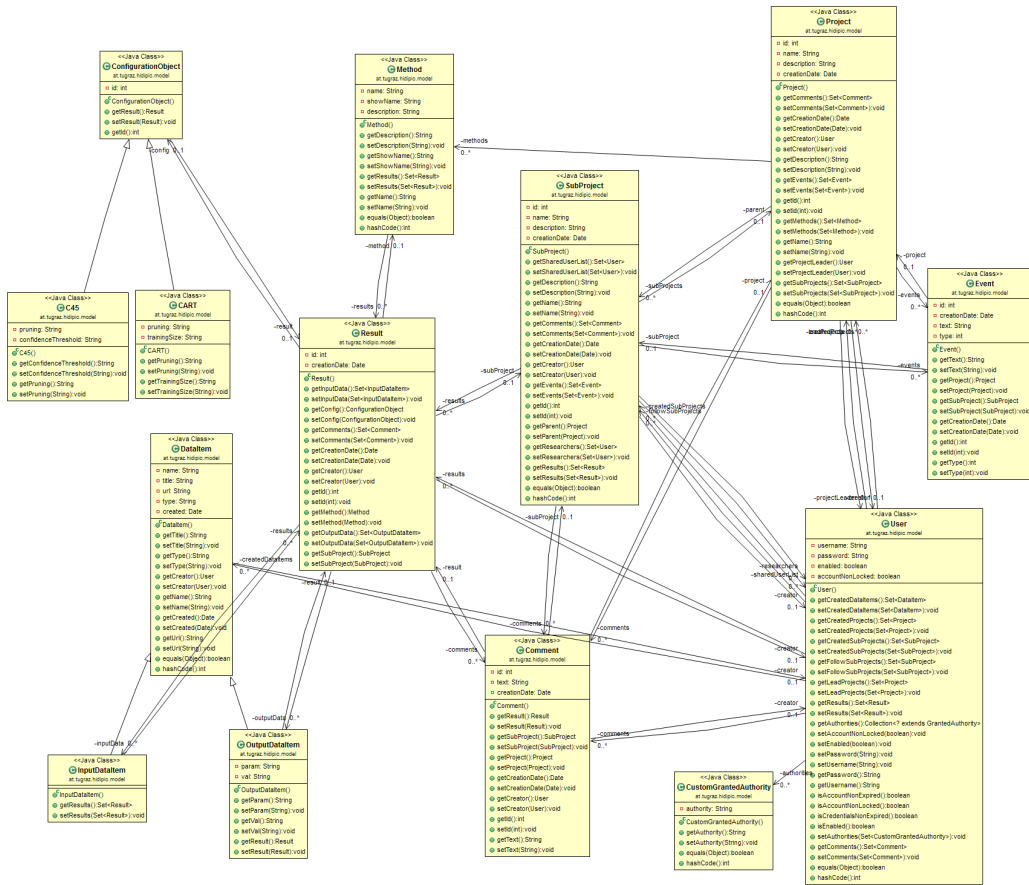


Figure 4.4: The basis for the Object Relational Mapping from which the data model is created.

4.2.2 Program structure

While the data model describes the underlying objects, the intended workflow and interaction between these entities has to be implemented, which in the case of an MVC-based application happens in the controller part of the software. In the KN-ODWAT framework, there are four different controller classes based on the particular functionality they provide and, moreover, based on the different authorization requirement necessary to access their services. Figure 4.6 represents these four controller classes, their interactions between each other and external utility objects. The majority of the methods within these controllers are request mappings. That means, that they are annotated with the `@RequestMapping` annotation, specifying the type of request and the URL the method is mapped to. Within the Spring framework, this is usually done by returning a `ModelAndView` object, containing data and information on where to find the frontend source for the response, for example a `.jsp` file. Figure 4.5 is an example of such a request mapping, showing the method which responds to a `GET` request on the URL `/allresults` within the application and returns the view `results`, referring to the `results.jsp` file and the model containing all results, which were created by the user who was currently logged in.

```
@RequestMapping(value = "/allresults", method = RequestMethod.GET)
public ModelAndView showAllResultsForUser() {
    ModelAndView mav = new ModelAndView("results");
    User loggedIn = (User) SecurityContextHolder.getContext().getAuthentication().getPrincipal();
    User user = hibernateTemplate.get(User.class, loggedIn.getUsername());
    List<Result> createdResults = dataAccessService.getCreatedResultsForUser(user, null);
    mav.addObject("lastResults", createdResults);
    return mav;
}
```

Figure 4.5: `DataAccessService` with the different controller classes using it.

Most of the methods within these controllers have similar functionality. They retrieve model data based on specified parameters from the database and return a suitable view with this data back to the user. Complex queries and data requests are handled by the `DataAccessService`, which is a crucial part of the KNODWAT's controller module. This service resides in the center of figure 4.6, because it is the main interface between the application and the underlying data model. At this point it is worth mentioning, that the deletion of data is deactivated in the current version of the software, in order to retain data consistency. More on this topic is discussed in chapter 7

As mentioned above, there are four controller classes:

- *PageController*

This controller class provides functionality available to every user of the application, such as displaying the landing page, help page, subprojects, projects, events, login, logout, settings and results. There is no general authorization requirement for these parts of the application.

- *AdminController*

As the name suggests, this controller handles the backend of the administration panel, making it possible for administrators to create, edit and delete users, projects, subprojects and methods. Only users with administrative rights have access to these methods.

- *ResearcherController*

This controller handles requests, which can be accessed by researchers. It includes project- and subproject management as well as file management.

- *CreateController*

Being responsible especially for critical tasks related to creating new results, which can only be performed by researchers, this controller has been split from the *ResearcherController*, because the functionality behind the creation of results is subject to a higher level of extension and change over time

Another crucial part of the application structure within KNODWAT is the result creation module represented in figure 4.7. The architecture of this module is optimized for extension. The general idea behind the extensibility features of KNODWAT was, to include different algorithms using the strategy design pattern. This highly popular design pattern is based on the concept, that different algorithms used for the same task are encapsulated and made interchangeable, which works by using the same general interface for each of the implemented methods. In the case of KNODWAT, the *MethodExecution* interface and its implementation, the *DefaultMethodImpl* represent the basis of each algorithm. This is not an exact application of the strategy pattern, due to the different strategies not only implementing the interface, but extending the default method. This decision was made in order to increase the simplicity of creating a new algorithm, as the default method already provides some built-in methods for result-generation and event-firing, which can be used by all the different strategies, without the need of any other interacting class. In order to extend the framework by another algorithm, the developer only needs to create one class, thus extending *DefaultMethodImpl*, which handles the input data, performs the algorithm and calls the result generation method. This process has also been implemented using the concept of convention over configuration, meaning that as long as the developer uses the name of the method for the class implementation and the view, no further configuration is needed and the framework finds the suitable components on its own.

The default algorithm is capable of firing three types of events, namely: start, end and error. Each of these events can be enriched by a string containing a reason for the error or useful information regarding the algorithm. By calling the *createResult* with a list of created output data items, the framework creates all necessary objects, links them together and redirects the user back to the landing page.

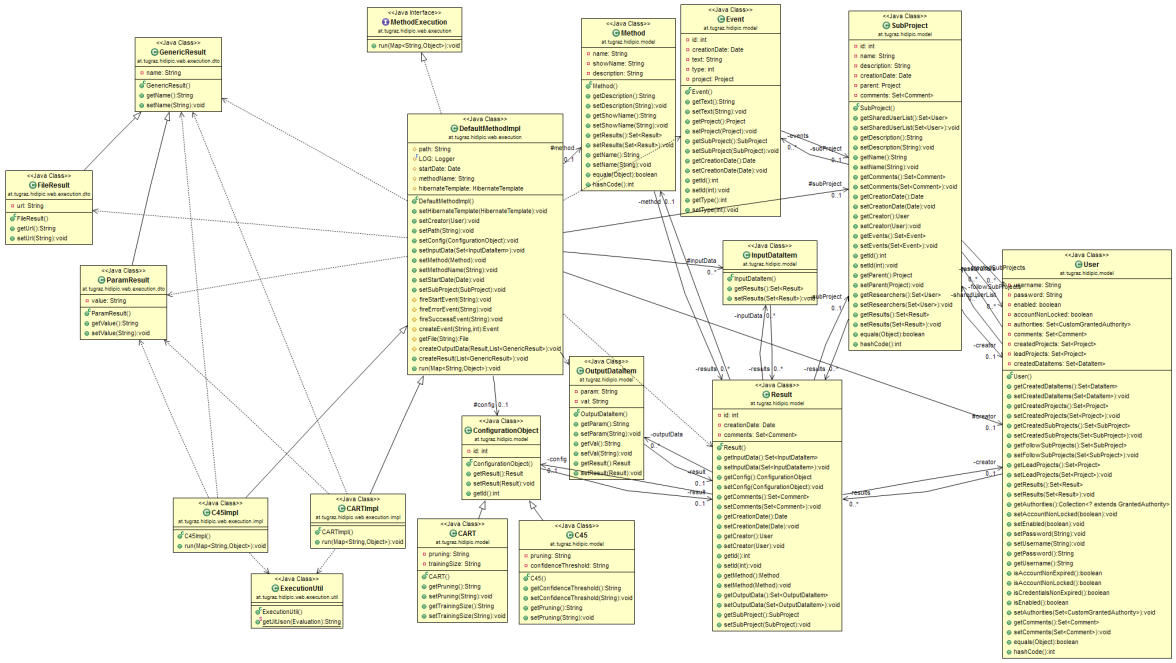


Figure 4.7: Class diagram of the method execution module

4.2.3 Frontend

The frontend of the KNODWAT platform, or the view part of the MVC-structure (Holzinger et al., 2010), consists mostly of JSP files. These Java Server Pages are, together with a model containing data to be visualized, returned to the server, which subsequently transforms them to HTML, ready to be viewed by the user. Figure 4.8 shows the JSP files used, structured by application hierarchy.

Mostly, the mentioned views contain HTML markup with CSS styling and JavaScript widget code. The model data is displayed using the JSTL - the Java Server Pages Standard Tag Library, which provides functionality for different data interactions such as formatting and iterating a collection of data. There are also Spring based tag libraries, such as the Spring Security tag library, making it possible to restrict user access in the front end as well. Tag libraries are a very comfortable tool for displaying and manipulating model data and are heavily utilized throughout the view part of the platform.

Java Server Pages also provide the possibility to include files, such as the *menu.jsp*, or the *actionbar.jsp*, which are used in a lot of different views. The two custom result views, *C45.jsp* and *CART.jsp*, are special cases where JSP files are included in KNODWAT. These files are included in the general *result.jsp*, and are created specifically for each implemented algorithm, in order to enable developers to create suitable visualizations of the generated data.

Spring MVC uses a so called *viewResolver* to connect the names of a view, which are specified in a controller to a JSP file. Frontend resources, such as JavaScript files, image and CSS stylesheets are also automatically mapped to the correct folder using Spring configuration, making it possible to use the same path to a resource file regardless of the current view's location.

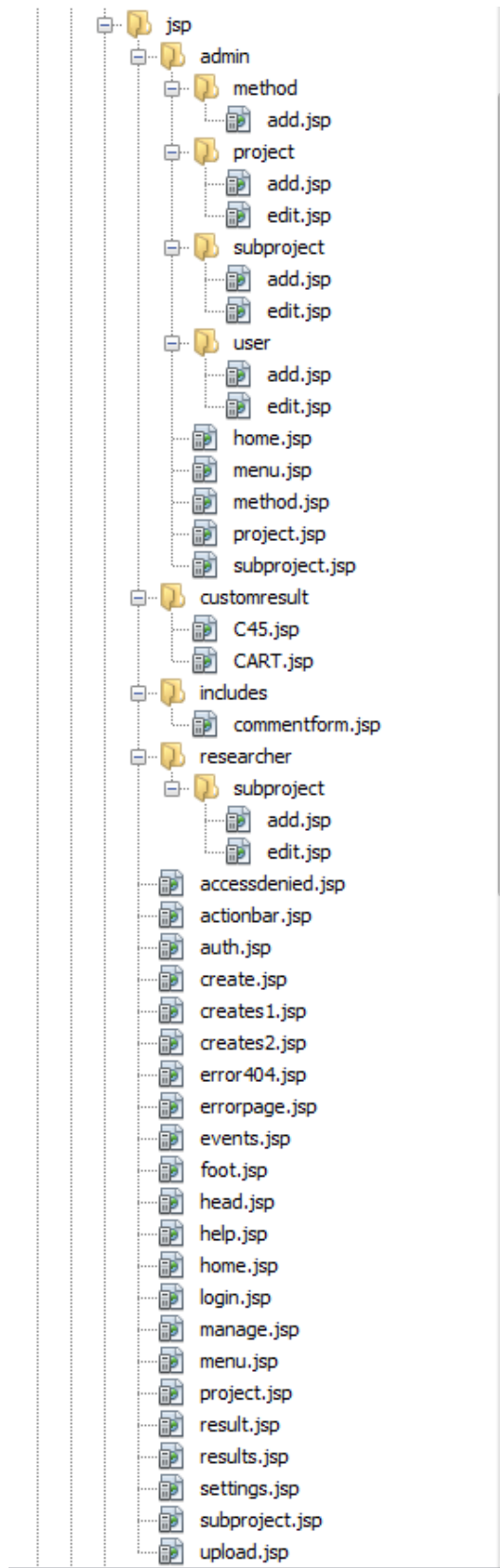


Figure 4.8: Structure of JSP-files (views) within the KNODWAT application.

4.2.3.1 Forms

The screenshot shows a web interface for editing a project. At the top, there is a navigation bar with links: Home, Upload, Create, Manage, Help, Logout. Below this, a secondary navigation bar highlights 'Projects' and includes 'Users', 'Subprojects', and 'Methods'. The main content area is titled 'EDIT DETAILS FOR CLASSIFICATIONDEMO'. It contains a form with the following fields: 'Name' (text input with value 'ClassificationDemo'), 'Description' (text area with value 'This is a project for demo purposes. It can be used to test two different classification-'), 'Project Leader' (dropdown menu with value 'admin'), and 'Methods' (multi-select input with values 'C45' and 'CART'). At the bottom of the form are 'Save' and 'Reset' buttons. On the right side, there is a sidebar with 'admin Logout' and 'ACTIONS' (Settings, Administration).

Figure 4.9: Form for editing an existing project

Many of the user interactions within the KNODWAT application happen via HTML forms. These forms are created using the Spring form tag library, which enables automatic model binding and validation, making it very comfortable to use. Most of the form elements used in the web interface are fairly basic, but the standard select and multi select elements of HTML, which are used for crucial data interactions within the application had to be improved due to the lack of their usability.

Select & Multiselect

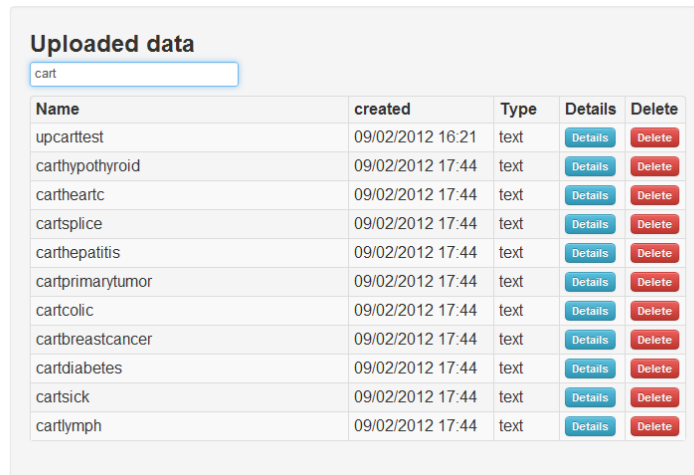
For the improvement of the usability regarding select and multi select fields, the Chosen plugin was established within KNODWAT. This plugin changes the style and interactive features of these elements immensely, making them very intuitive to use. An example is presented in figure 4.10

The screenshot shows a web interface for adding a subproject. At the top, there is a navigation bar with links: Home, Upload, Create, Manage, Help, Logout. Below this, a secondary navigation bar highlights 'Subprojects' and includes 'Users', 'Projects', and 'Methods'. The main content area is titled 'ADD SUBPROJECT CART'. It contains a form with the following fields: 'Name' (text input with value 'CART'), 'Description' (text area with value 'Demo-Project for CART'), 'Researchers' (multi-select input with values 'admin', 'res_pro1', 'res_pro2', 'sample_res'), and 'Parent Project' (dropdown menu with value 'ClassificationDemo'). At the bottom of the form are 'Save' and 'Reset' buttons. On the right side, there is a sidebar with 'admin Logout' and 'ACTIONS' (Settings, Administration).

Figure 4.10: Multiselect widget in the forms of KNODWAT

4.2.3.2 Filters

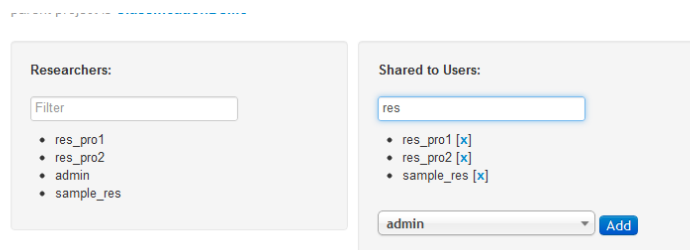
Another usability feature for navigating the content of the web interface more efficiently are jQuery-based content filters for lists and tables. These filters allow the user to limit the amount of data shown in a table view or list view, by providing a search term. Examples for this behavior are presented in figure 4.11 and figure 4.12.



The screenshot shows a web interface with a search filter and a table. The search filter is labeled 'Uploaded data' and contains the text 'cart'. Below the filter is a table with the following data:

Name	created	Type	Details	Delete
upcarttest	09/02/2012 16:21	text	Details	Delete
carthyothyroid	09/02/2012 17:44	text	Details	Delete
cartheartc	09/02/2012 17:44	text	Details	Delete
cartsplice	09/02/2012 17:44	text	Details	Delete
carthepatitis	09/02/2012 17:44	text	Details	Delete
cartprimarytumor	09/02/2012 17:44	text	Details	Delete
cartcolic	09/02/2012 17:44	text	Details	Delete
cartbreastcancer	09/02/2012 17:44	text	Details	Delete
cartdiabetes	09/02/2012 17:44	text	Details	Delete
cartsick	09/02/2012 17:44	text	Details	Delete
cartlymph	09/02/2012 17:44	text	Details	Delete

Figure 4.11: Table filtering via JavaScript.



The screenshot shows two panels. The left panel is titled 'Researchers:' and has a search filter. Below the filter is a list of items:

- res_pro1
- res_pro2
- admin
- sample_res

The right panel is titled 'Shared to Users:' and has a search filter. Below the filter is a list of items:

- res_pro1 [x]
- res_pro2 [x]
- sample_res [x]

Below the list is a dropdown menu with 'admin' selected and an 'Add' button.

Figure 4.12: List filtering via JavaScript.

4.2.4 Functionality

This section provides an overview of the different views within KNODWAT and their underlying functionality using screenshots.

4.2.4.1 Home, Help & Error page

After successfully logging in, the user is redirected to the landing page. If the user is not currently following any subprojects, the landing page only consists of a table showing certain statistics about the framework usage, such as the number of projects, subprojects, results, users and events as shown in figure 4.13.

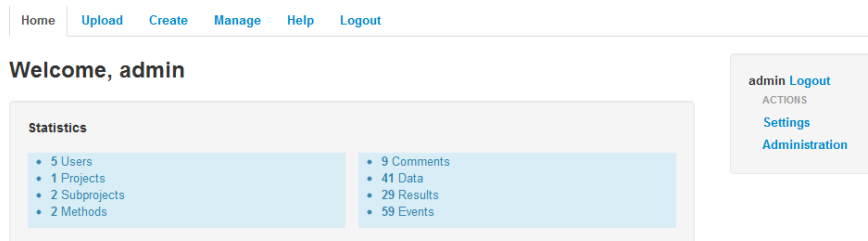


Figure 4.13: Landing page without following subprojects, containing only statistics about the database.

If the currently logged in user, however, follows subprojects, the landing page changes in order to provide both a fast way to these subprojects of interest as well as a list of recent activities within them, by showing an event feed. An example for this changed behavior is displayed in figure 4.14.

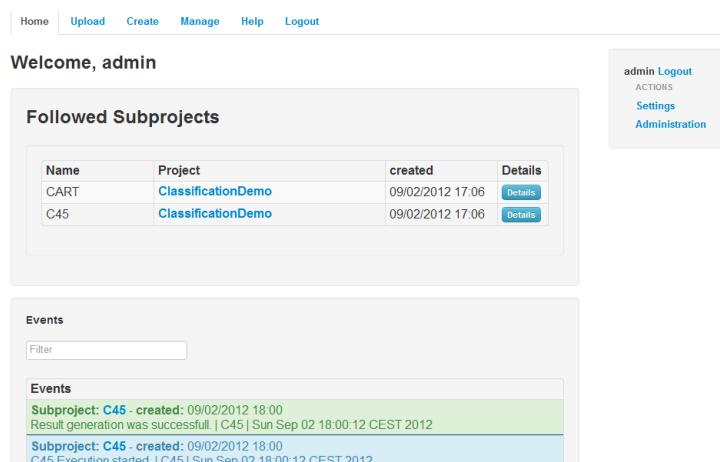


Figure 4.14: Landing page when following subprojects, with an overview of the projects and recent events within them.

To introduce new users to the functionality that the KNODWAT framework has to offer, there is a help site available. The content of the help site change according to the logged in users authority level, meaning that an administrator has access to the documentation regarding the administration panel, the researchers area and the general users area, whereas a normal user is only able to see the content describing the users area. A part of this site is presented in figure 4.15.

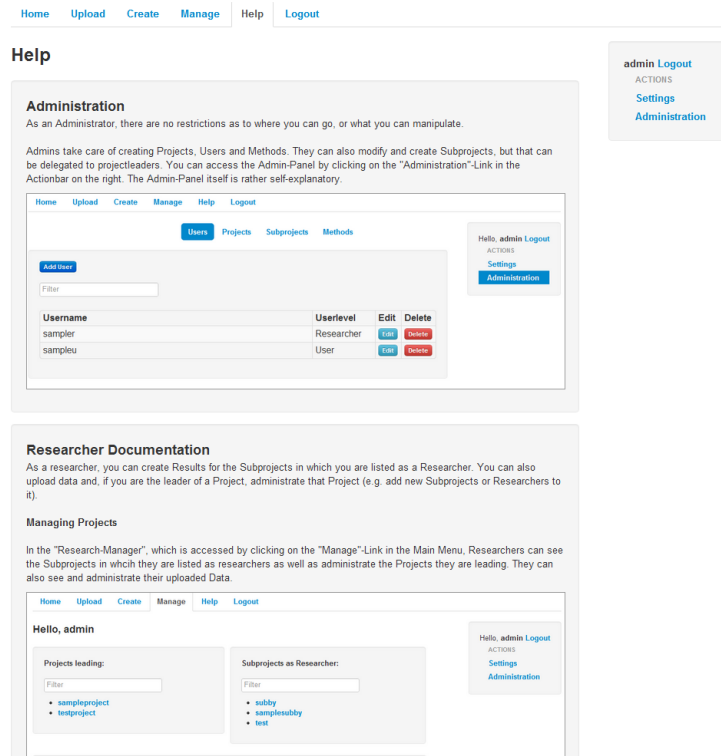


Figure 4.15: The Help page, to introduce new users to the system.

In order not to confuse users as well as not to reveal any critical information about the insides of the application, every exception and error leads to the error page by default depicted in figure 4.16.

An error has occurred!
[return](#)

Figure 4.16: Default error page.

4.2.4.2 Administration

The administration panel allows an administrator to manage users, projects, sub-projects and methods. In the case of methods, this functionality is only used if an existing method has to be deleted, or if the framework has been extended by a new algorithm, which subsequently has to be added to the database with the correct name, thus enabling the convention over configuration methodology behind the process of method execution.

Users can only be created by the administrator, which means that there is no way for new users to register themselves. This functionality can of course be extended, in order to make the framework better accessible to the public, but for the default case, where only a few selected members of a research community and students work within an instance of the framework, the method that is currently in effect should be sufficient. There are three user roles currently in place:

- Administrator

Usually only one user has this role. The administrator is responsible for managing users and creating projects. When creating a project, administrators should choose a researcher to lead the project and then manage it in the future. There are no restrictions for an administrator in place.

- Researcher

Researchers are the main user group in the KNODWAT framework, they manage their subprojects and projects if they are their leader, create input data and generate results. Researchers can see their own projects, subprojects and results, as well as the ones shared with them.

- User

Every user has this role. Users can only see projects, subprojects and results if they are explicitly shared with them and they are following them. Every user can write comments on content that he or she can see.

Figures 4.17, 4.18, 4.19 show some examples of views in the administration panel.

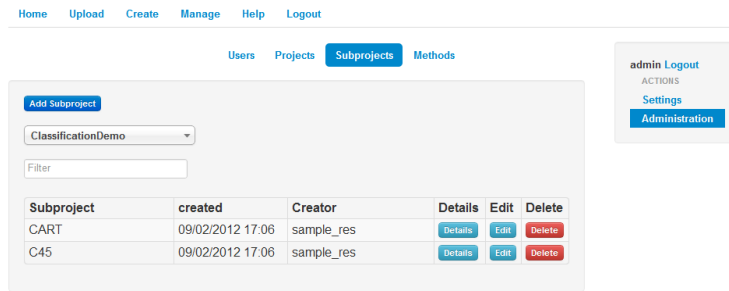


Figure 4.17: Overview of the subprojects within the administration panel

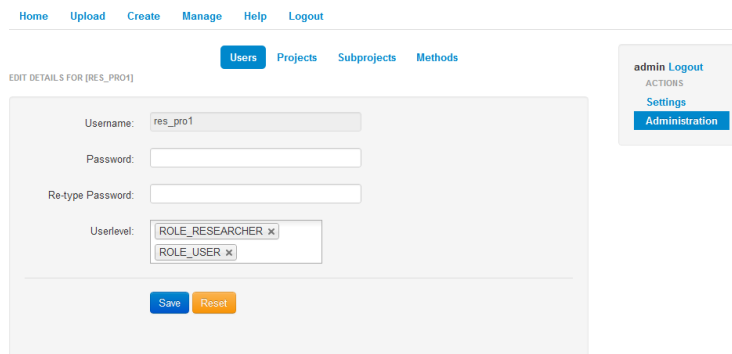


Figure 4.18: Editing a user.

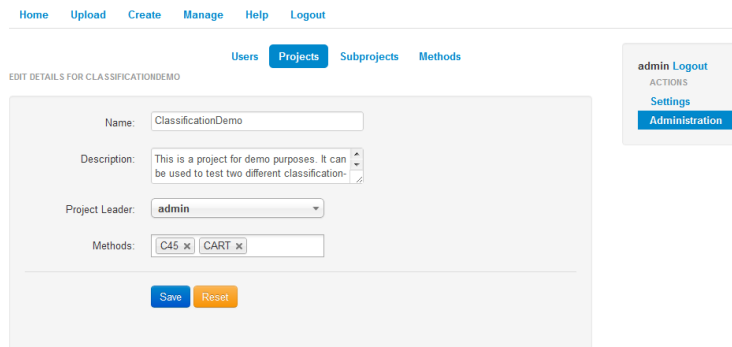


Figure 4.19: Creating and editing a project inside the admin panel.

4.2.4.3 Result creation

The result creation or method execution functionality of KNODWAT is arguably the core of the application, considering that the main purpose of this software is to do research by generating data, by applying different algorithms to various data sets. The procedure is rather simple, as depicted in figure 4.20. At first the researcher has to select the project for which he or she wants to create a new result. Then, in step two, the subproject and the method are chosen from within the selected project as shown in figure 4.21. Obviously, only projects, subprojects and methods available to the logged in researcher are shown in this selection procedure. Finally, the user proceeds to a result creation page as presented in figure 4.22, where the chosen project, subproject and method are stated. Additionally, a short documentation on how to use the chosen method is displayed and the researcher can choose input data for the algorithm from his or her collection of uploaded files. Then, method-specific parameters are provided by the user and the execution can start.

In the background, the application uses the name of the method to find the configuration, providing method-specific parameters for the chosen algorithm. Afterwards, the concrete implementation of the algorithm (e.g. *CARTImpl*) is instantiated using the Java reflection functionality and the method name and finally the *run()* method is executed, starting the result generation process.

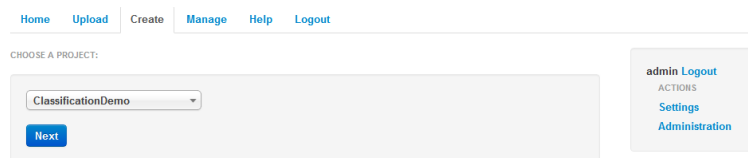


Figure 4.20: Selection of the project, within the result will be created.

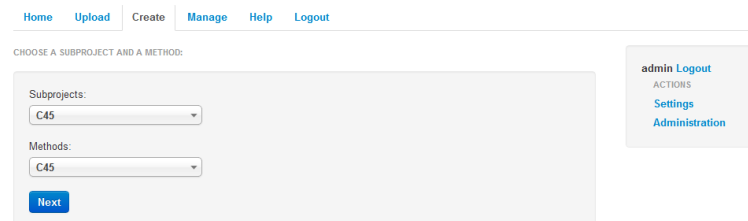


Figure 4.21: Selection of a subproject and the method for the result.

Home Upload Create Manage Help Logout

USE METHOD

Method: C45

Project: ClassificationDemo

Subproject: C45

Description

This C4.5 implementation takes two ARFF Files.

- * One File has to be named 'train', this is the training set
- * The second file has to be named 'test', this is the data set the trained classifier will be tested on.

The 'ConfidenceThreshold'-parameter specifies the confidence threshold for pruning, if pruning is activated, it is 0.25 by default. (0.0 - 0.5)

The 'Pruning'-parameter specifies if the algorithm should perform pruning or not. It does not do so by default. (y or n)

Input Data:

Filter

Name	Identifier	created	Name
<input type="text"/>	c45diabetestest	09/02/2012 17:46	<input type="checkbox"/>
<input type="text"/>	c45hypothyroidtrain	09/02/2012 17:46	<input type="checkbox"/>
<input type="text"/>	c45colicestest	09/02/2012 17:46	<input type="checkbox"/>
<input type="text"/>	upcarttest	09/02/2012 16:21	<input type="checkbox"/>
<input type="text"/>	c45colicestest	09/02/2012	<input type="checkbox"/>

admin Logout
ACTIONS
Settings
Administration

Figure 4.22: Specifying parameters and input data for the result creation.

4.2.4.4 Project management

For researchers, the *Manage* menu option, provides an overview of the projects, which are led by the current user, the subprojects where the particular user is stated as a researcher, a list of the most recently generated results, an administration panel for the projects managed by the user and a table showing the user's uploaded files as depicted in figures 4.23 and 4.24.

Using this management panel, a researcher can keep an overview of recent activities and available data, as well as make use of the fast ways of creating, editing and deleting subprojects or assigning new researchers to the projects led by him or her. The functionality in the background is fairly simple with normal CRUD operations and user access restrictions embedded in order to avoid the manipulation of foreign projects.

Home Upload Create Manage Help Logout

Hello, admin

admin Logout
ACTIONS
Settings
Administration

Projects leading:

Filter

- ClassificationDemo

Subprojects as Researcher:

Filter

- CART
- C45

Results

Filter

creation Date	Method	Creator	Details
09/02/2012 18:00	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:57	C45	admin	Details
09/02/2012 17:57	C45	admin	Details
09/02/2012 17:57	C45	admin	Details
09/02/2012 17:57	C45	admin	Details
09/02/2012 17:56	C45	admin	Details
09/02/2012 17:56	C45	admin	Details

[Show all Results](#)

Subprojects of Projects lead

ClassificationDemo

Name	created	Details	Edit	Delete
CART	09/02/2012 17:06	Details	Edit	Delete
C45	09/02/2012 17:06	Details	Edit	Delete

[Add Subproject](#)

Figure 4.23: Project management overview part 1

09/02/2012 17:57	C45	admin	Details
09/02/2012 17:56	C45	admin	Details
09/02/2012 17:56	C45	admin	Details

[Show all Results](#)

Subprojects of Projects lead

ClassificationDemo

Name	created	Details	Edit	Delete
CART	09/02/2012 17:06	Details	Edit	Delete
C45	09/02/2012 17:06	Details	Edit	Delete

[Add Subproject](#)

Uploaded data

Filter

Name	created	Type	Details	Delete
upcarttest	09/02/2012 16:21	text	Details	Delete
carthyothyroid	09/02/2012 17:44	text	Details	Delete
carthearc	09/02/2012 17:44	text	Details	Delete
cartsplice	09/02/2012 17:44	text	Details	Delete
carthepatitis	09/02/2012 17:44	text	Details	Delete
cartprimarytumor	09/02/2012 17:44	text	Details	Delete
cartcolic	09/02/2012 17:44	text	Details	Delete
cartbreastcancer	09/02/2012 17:44	text	Details	Delete

Figure 4.24: Project management overview part 2

4.2.4.5 File-Upload

Researchers have the ability to upload files as input data for the method execution by using the integrated Blueimp jQuery File Upload widget as shown in figure 4.25. The widget is extremely easy to use, providing color coded buttons, progress bars and the ability to upload multiple files at once. The user has to provide a specified title for each uploaded file in order to make it easier to find and identify files after their upload. The title is not a unique attribute and can be chosen freely. In the background, however, each file is assigned a random identifier using UUID - Universally unique identifier, both for the file name and the identifier in the database, in order to avoid conflicts between the uploaded data of different researchers. The upload system automatically recognizes, if the file is an image, a CSV-file or a simple text file and flags it accordingly. Using the default configuration, the limit of a file selected for uploading is 1 MB, but this can be changed easily. There is currently no restriction on any file types, but the framework is already equipped with functional methods for invoking such restrictions if necessary at some point.

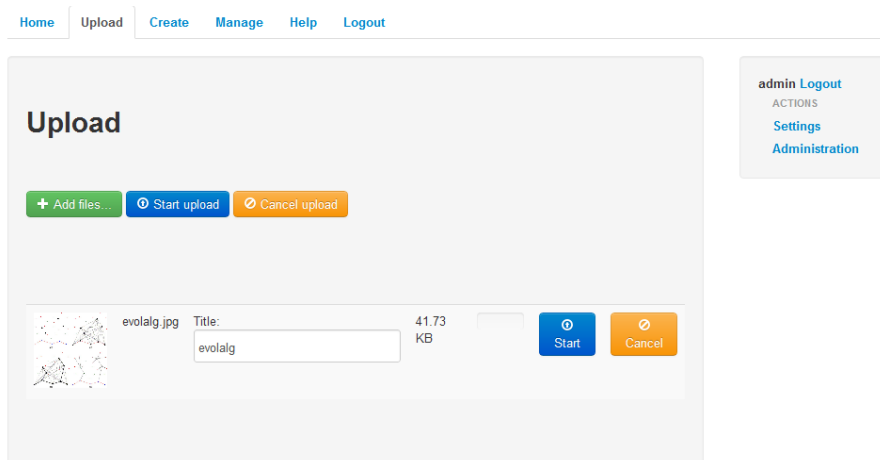


Figure 4.25: Uploading a file, to create input data.

4.2.4.6 Follow, Share & Comments

In addition to its core functionality of providing a manageable environment for research and result generation, KNODWAT also comprises a few social features, which enable users to share and follow as well as comment on projects, subprojects and results. Figure 4.26 shows the *Settings* page, accessible by clicking on the link in the sidebar, which provides users with the ability to change their password, the language and, most importantly, to follow or unfollow subprojects shared with them.

Subproject	created	Follow
CART	09/02/2012 17:06	Follow
C45	09/02/2012 17:06	Unfollow

Figure 4.26: Settings page with follow and unfollow options.

In every detail view of a project, subproject and result there is a comment-panel, where users can write comments and read the comments other users have submitted on the topic at hand as shown in figure 4.27. Every user can comment on content shared with them. At the moment, comments can neither be edited or deleted. This restriction is part of the design, as the software will not be publicly accessible anyway and user accounts have to be created by the administrator, meaning that spam will not be an issue and that discussion is always a welcome addition when it comes to interpreting scientific data. However, this functionality, could be added within a short amount of time, if necessary, as the required infrastructure already exists.

Add Comment:

Text:

Comments:

- sample_user**, 09/02/2012 17:06
This is a test comment.
- admin**, 09/18/2012 20:16
another test comment
- admin**, 09/18/2012 20:16
commenting on the comment!
- sample_user**, 06/12/3912 00:00
This is also a test comment.

Figure 4.27: Commenting on results, projects and subprojects is possible within the KNODWAT application.

4.2.4.7 Event system

The event system within KNODWAT serves two purposes. On the one hand, it keeps users who are following a subproject up to date on its research progress and on the other hand, it provides notifications for researchers regarding their created results, showing errors and messages indicating the success or failure of the executed method. Events are, at the moment, only activated for subprojects, but this functionality can be easily extended to other objects such as projects or even users by making use of the infrastructure available in the framework.

On the frontend side, there are views showing the 10 most recent events, color coded in blue for start events, green for end events and red for error events. Under these lists, there is always a link to a view containing all events of the current subproject or user. Events are fired from within the actual implementations of the algorithms, implying that the question which events occur at what time depends on when they are fired within the *run()* method. Figure 4.28 depicts an example list of events, which occurred during the course of creating results within the C45 subproject.

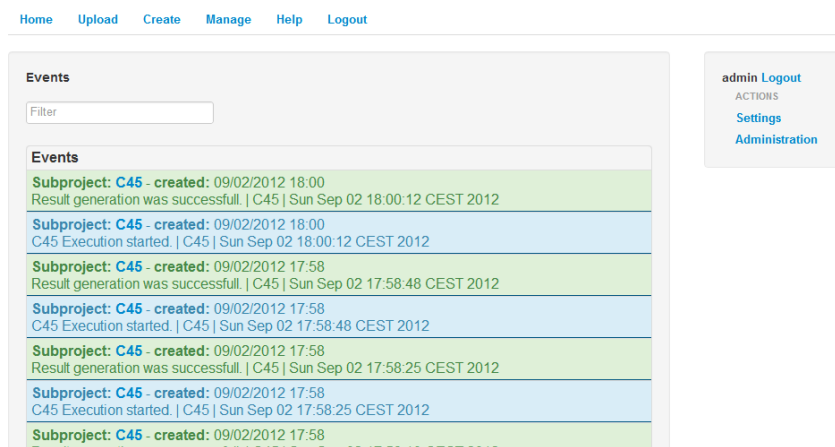


Figure 4.28: Overview of recent events happening in related projects.

4.2.5 Technical Details

In this section, some of the more complex technical parts inside the KNODWAT application are described, such as the dynamic handling of method attributes, the localization of the web application and the user authentication among others.

4.2.5.1 Dynamic method attributes

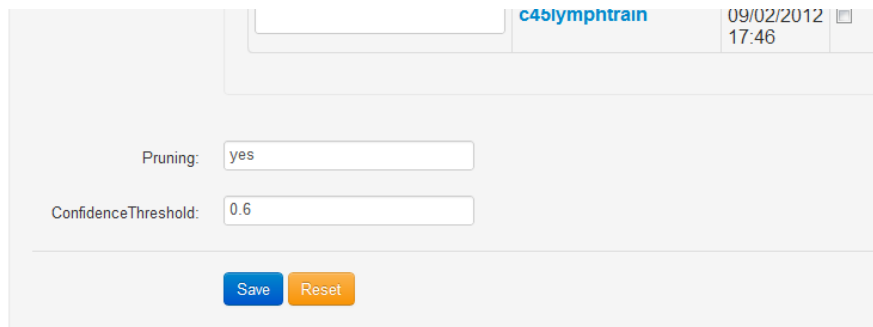
Extensibility is one of the key features of the KNODWAT application. In order to make it easier to extend the framework with different algorithms, each method can specify its own set of custom parameters. In this software the process is handled by using a custom Java annotation called *FormElement*, which allows the developer to annotate the properties of a configuration object with parameters as depicted in figure 4.29. In its current version, only parameter identifiers and their display names can be configured. However, this custom annotation can be extended by, for example, adding a parameter for rendering in order to specify that this element, as an input element, is rendered as a checkbox. At the moment, only textual parameters can be used, but due to the fact that the actual method implementations are aware of this fact, this circumstance does not pose a problem, as input parameters can easily be parsed based on the underlying constraints.

```
public class CART extends ConfigurationObject {  
  
    @FormElement(fieldName = "Pruning", identifier = "pruning")  
    private String pruning;  
    @FormElement(fieldName = "TrainingSize", identifier = "trainingsize")  
    private String trainingSize;  
  
    public String getPruning() {  
        return pruning;  
    }  
  
    @FormElement(identifier = "pruning")  
    public void setPruning(String pruning) {  
        this.pruning = pruning;  
    }  
}
```

Figure 4.29: Configuration object annotated using the *FormElement* annotation.

These custom parameters are then displayed both in the result creation view and in the result detail view using the two custom JSP tags *DynamicConfigForm* and *DynamicConfigOutput*. The functionality behind these two tags is quite similar, the only difference being, that the former one displays HTML elements, while the latter one simply displays a list of the custom parameters. Technically, this works

using Java object reflection. The tag is initialized with a *ConfigurationObject* as input. This input is then analyzed based on its actual class (e.g. *CART*) and each of its properties and setter methods are checked for annotations. During this process, all properties and setter methods with an annotation of the type *FormElement* are added to a HTML markup, which is then returned to the view, making it possible to display the custom parameters of different methods using the same underlying view and thus greatly simplifying the process of framework extension. This method is also used in the *CreateController*, to make sure that the custom parameters are passed on to the correct method implementation. Figure 4.30 shows the output of the *DynamicConfigForm* tag in the method execution view.



c45lympntrain		09/02/2012 17:46
Pruning:	<input type="text" value="yes"/>	
ConfidenceThreshold:	<input type="text" value="0.6"/>	
<input type="button" value="Save"/> <input type="button" value="Reset"/>		

Figure 4.30: Custom parameters displayed for every different method.

4.2.5.2 Custom tag library

In order to simplify some of the data visualization, as well as to implement more complex functionality in the different views, the *KDDTagLibrary* was added to the KNODWAT project. This library is a small, custom built collection of tags and functions providing some basic, yet powerful functionality needed in some views. The library consists of the following functions as well as the two tags mentioned in 4.2.5.1:

- *isActiveMenu*

It is a function that uses the current URL string and the name of the different menu options as input and returns a boolean value for the active menu option, in order to show the user visually, in which section he or she is currently active.

- *isInstanceOf*

It is basically the same function as the normal Java *instanceof*, checking if the tested class is a type of the specified class. This function is used to dynamically check, if an output data object is a file or a string, in order to render the two types differently.

- *hasRole*

This function is used for frontend user restriction, checking if the currently logged in user has the tested user role assigned to him or her.

- *getUserRole*

Similar to *hasRole*, this method returns the name of the current user's highest user role.

- *isFollowProject*

This function tests, if the subproject at hand is being followed by the logged in user, in order to render only projects the user is actually interested in.

- *nl2br*

Similar to the PHP function, it replaces line breaks with HTML line breaks.

- *getTypeColor*

This function basically converts an Integer to a String based on an underlying mapping.

4.2.5.3 Localization

Spring framework provides a rather simple way to localize MVC-applications to other languages. For the sake of demonstrating this feature, the KNODWAT platform has been localized for German, as shown in figure 4.31. The localization to other languages can be performed quite easily within a few hours if necessary.

Home Upload Erstellen Verwalten Hilfe Ausloggen

Hallo, admin

admin Ausloggen
AKTIONEN
Einstellungen
Administration

Projekte als Projektleiter:

Filter

- ClassificationDemo

Subprojekte als Forscher:

Filter

- CART
- C45

Resultate

Filter

Erstellungsdatum	Methode	Ersteller	Details
09/02/2012 18:00	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:58	C45	admin	Details
09/02/2012 17:57	C45	admin	Details
09/02/2012 17:57	C45	admin	Details
09/02/2012 17:57	C45	admin	Details

Figure 4.31: German localization is supported within the KNODWAT application.

4.2.5.4 Form validation

Spring framework provides extensive functionality for automatic form validation using the Spring form tag library and either custom validators or the `@Valid` annotation, which again uses the `javax.Validation` framework annotations such as `@Length` and `@Pattern`, to add constraints to properties. An example of this form validation is shown in figure 4.32. The current version of KNODWAT only supports very basic form validation, but this functionality can be extended quite easily if necessary. The current form validation also only works on the backend side of the web application. There is no JavaScript based frontend validation for inputs, but there are automatic ways, using annotations, to create such constraints within the Spring framework.

Figure 4.32: Basic form validation where no special characters and length limits are in place.

4.2.5.5 Authentication

When using Spring Security to extend an existing Spring based application with user authentication and restriction features, the amount of actual Java code that has to be written is surprisingly low. Most of the functionality is based on of simple configuration, where the developer can tell the Spring Security extension which way of checking user data to use (in this case database based), as well as provide simple interception mappings, restricting existing user roles to certain areas by using URL patterns as shown in figure 4.33.

When a user navigates to the home site of KNODWAT, a login screen is presented (figure 4.34). The whole process of logging in, redirecting the user and handling authentication errors can be configured using 3 lines of XML. The only actual Java code necessary for user authentication is inside the *CustomUserDetailsService*, which basically just overwrites the *loadUserByUsername* method of the Spring Security *UserDetailsService* interface. In the case of KNODWAT, a user gets loaded via database query, which means that the *User* entity has to implement the *UserDetails* interface, in order to use all the built-in features of Spring Security.

```
<security:http pattern="/resources/**" security="none" />

<security:http auto-config="true" use-expressions="true">
  <security:intercept-url pattern="/login" access="permitAll"/>
  <security:intercept-url pattern="/admin/**" access="hasRole('ROLE_ADMIN')"/>
  <security:intercept-url pattern="/manage" access="hasAnyRole('ROLE_RESEARCHER', 'ROLE_ADMIN')"/>
  <security:intercept-url pattern="/upload" access="hasAnyRole('ROLE_RESEARCHER', 'ROLE_ADMIN')"/>
  <security:intercept-url pattern="/upload/**" access="hasAnyRole('ROLE_RESEARCHER', 'ROLE_ADMIN')"/>
  <security:intercept-url pattern="/create" access="hasAnyRole('ROLE_RESEARCHER', 'ROLE_ADMIN')"/>
  <security:intercept-url pattern="/create/**" access="hasAnyRole('ROLE_RESEARCHER', 'ROLE_ADMIN')"/>
  <security:intercept-url pattern="/researcher/**" access="hasAnyRole('ROLE_RESEARCHER', 'ROLE_ADMIN')"/>
  <security:intercept-url pattern="/**" access="isAuthenticated()"/>
  <security:form-login login-page="/login" authentication-failure-url="/login?login_error=true" default-target-url="/home"/>
  <security:access-denied-handler error-page="/accessdenied"/>
  <security:logout invalidate-session="true" logout-success-url="/login" logout-url="/logout"/>
</security:http>

<bean id="customUserDetailsService" class="at.tugraz.hidpic.user.CustomUserDetailsService" init-method="initialize" >
</bean>

<security:authentication-manager>
  <security:authentication-provider user-service-ref="customUserDetailsService">
    <security:password-encoder hash="md5"/>
  </security:authentication-provider>
</security:authentication-manager>
```

Figure 4.33: Spring Security config, showing user roles and their restrictions.

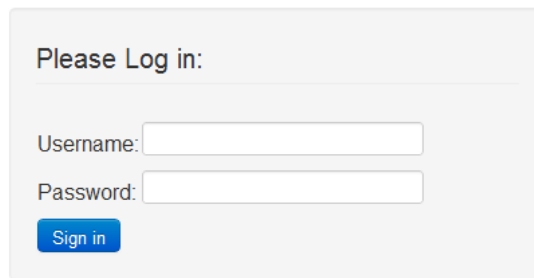


Figure 4.34: The login screen.

If a user is not allowed to visit a certain URL (e.g. a non-researcher clicking on the *Upload* menu option), the access denied page is displayed as shown in figure 4.35.



Figure 4.35: If the user does not have sufficient rights to view a site, this error page is displayed.

4.2.5.6 Database transactions using Hibernate

In the KNODWAT application, the database transaction layer is based on the Hibernate framework integrated within Spring. Most queries are executed using the *HibernateTemplate*, a helper class provided by Spring framework for automated transaction and session handling, making it very simple to perform CRUD operations as well as to find specific data rows. One of the perks when using object relational mapping frameworks such as Hibernate is, that executed queries already return instantiated Java objects, ready for further processing and capable of class inheritance. Some simple queries, such as cases where a single data row from a single table has to be found, are usually executed directly where they are needed, by injecting the *HibernateTemplate*. More complex queries with multiple joins or data pre- and post processing are handled inside the *DataAccessService*.

4.2.5.7 Method implementations

In order to test the KNODWAT framework and demonstrate its extensibility, two knowledge discovery methods were implemented: C4.5 and CART. Both methods are very popular classification tree algorithms. The actual implementation of the two methods uses the WEKA library, which provides a comfortable way of using .ARFF files as an input for both algorithms. Due to WEKA being very well documented, the implementation of the algorithms is not a problem, however, the two methods are only implemented in a very basic way for demonstration purposes and do not contain all the features provided by the WEKA library. Both methods have two custom parameters with one of them being a modifier specifying if the resulting classification tree should be pruned or not. In the case of CART, the second parameter is a float value describing the percentage of input data, which should be used for training. The second parameter of the C4.5 implementation is the confidence threshold used for the execution. CART only uses one ARFF input file, while C4.5 uses two, one for training and one for testing.

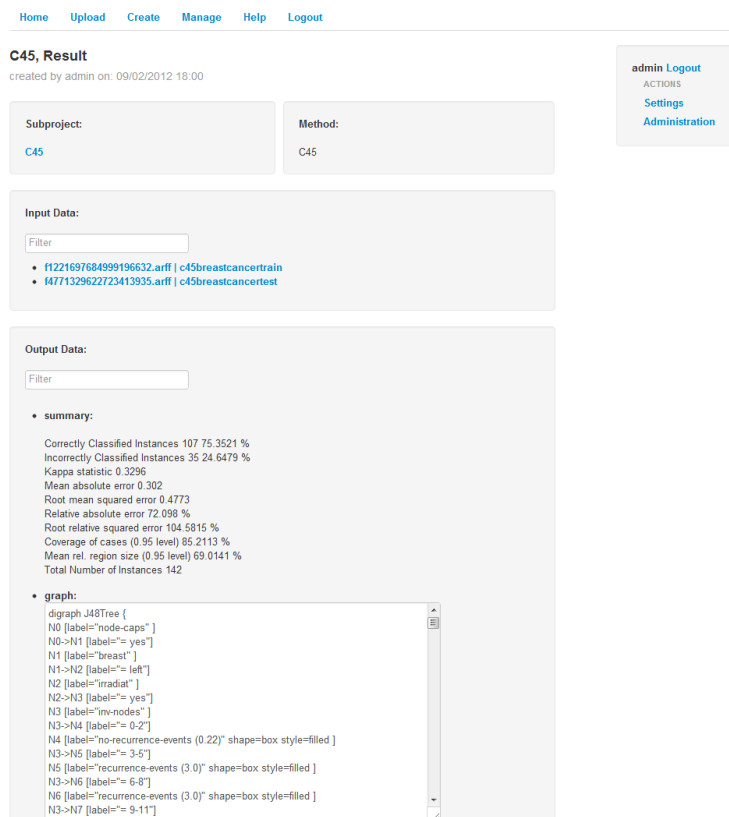


Figure 4.36: Result detail part 1

The result detail view of the two methods only differs in one detail, the C4.5 implementation provides a tree visualization in the form of a string, which can be visualized using GraphViz Dotty. Apart from that, the generated results are identical, providing an overview of the input data, parameter values and results in the form of a JavaScript InfoVis Toolkit graph of the correctly and incorrectly classified instances, statistics and a matrix as depicted in figures 4.36 and 4.37. Further information, providing reasons why these two methods were chosen and including a benchmark based on two different data sets can be found in chapter 5.

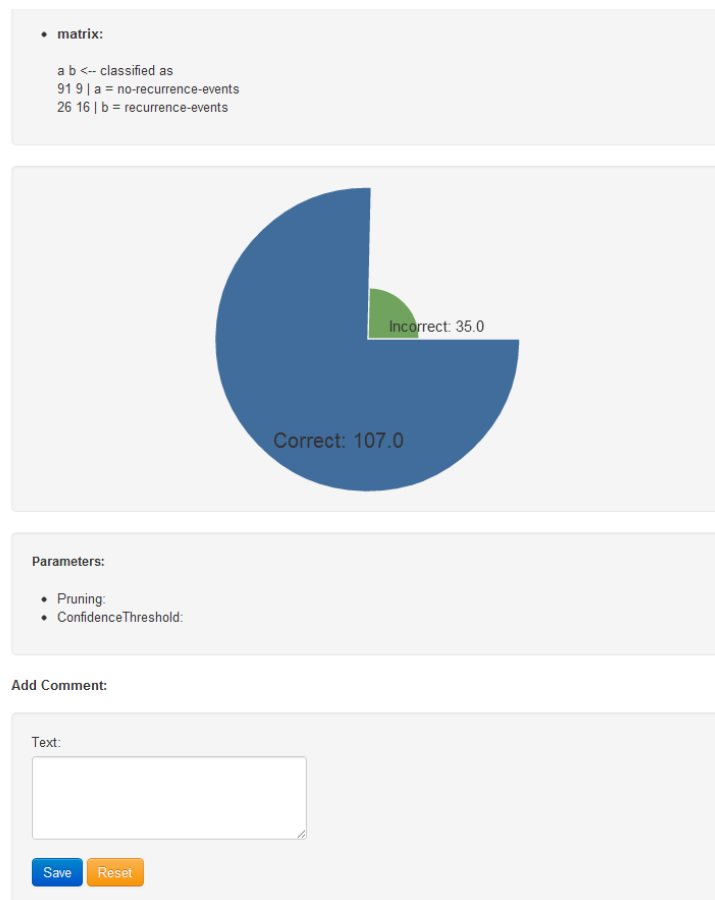


Figure 4.37: Result detail part 2

4.2.6 Testing

The KNODWAT application was tested empirically by multiple people with experience in the field of software engineering, knowledge discovery and web development. The developer tested the software intensively and colleagues provided a lot of useful feedback and bug reports. There was no formal usability testing due to time constraints. This topic is handled in more detail in chapter 7.

4.2.6.1 Unit Tests

The KNODWAT framework does not provide a test suite, which would be very useful especially for developers who are currently extending the framework yet. There are only three unit tests implemented at the moment, which are not for unit testing in the traditional sense, but are used as sandbox tests for the testing of new technology, such as the WEKA algorithms. Creating a test suite containing both unit and integration tests is definitely something worth looking at in the future, as described in chapter 7, but could not be realized in the given time frame. The three sandbox tests currently available, which can provide some useful pointers for developers in regard to adding new algorithms from the WEKA library, are the following:

- *CARTTest*
it is a simple implementation of the *SimpleCART* algorithm using WEKA. The method takes an ARFF file and produces different statistics as well as a JSON string.
- *C45Test*
This test is the same as the *CARTTest*, but for the C4.5 algorithm (J48 in WEKA)
- *ReflectionTest*
It is a small test exploring the functionality used to provide dynamic attributes using annotations and reflection.

4.2.7 Extensibility

In order to make the KNODWAT framework applicable in many different disciplines of science, it has to be easy to add new methods to the existing platform. Currently it is not possible to extend the framework without any software engineering experience, but considering the way in which the extensibility functionality of KNODWAT is built, even a rather inexperienced programmer with some knowledge of the Java programming language and the ability to follow a few simple, well documented steps, is able to add new methods to the application. This section describes these steps, providing a tutorial on how to add new algorithms to the KNODWAT application based on the example of adding CART - Classification and Regression Trees.

4.2.7.1 Configuration object

The first step is to create a new configuration object for the chosen algorithm. Such an object is displayed in figure 4.38. Basically, this object represents the custom parameters used for the algorithm, in the case of CART, one parameter will be created to enable the user to prune trees and another one to control how many data elements are used for training. It is important to note, that the extension of KNODWAT works by the use of convention over configuration, which means, that the naming of the created classes is relevant, as it will be used to link the created parts together afterwards. In the case of CART, everything will be named "CART", so the first step is to create a class named *CART* in the *at.tugraz.hidipic.model* package, which is an extension of the *ConfigurationObject* and is annotated with the *@Entity* annotation. A table name, using the *@Table(name = "")* annotation can be supplied as well, but is not mandatory. The next step is to define the parameters as String objects within the class and to annotate them with the *@FormElement* annotation, thus supplying a *fieldName*, which is the name that will be displayed in views, and the *identifier*, which is used to identify the property throughout the application. In order to enable data interaction, setter and getter methods have to be created for each property and the setter methods have to be annotated with the *@FormElement* annotation as well, but here, only the *identifier* has to be specified, which has to have the same value as the corresponding property of the setter method. This concludes the creation of the configuration object.


```

@Entity
@Table(name = "cart")
public class CART extends ConfigurationObject {

    @FormElement(fieldName = "Pruning", identifier = "pruning")
    private String pruning;
    @FormElement(fieldName = "TrainingSize", identifier = "trainingsize")
    private String trainingSize;

    public String getPruning() {
        return pruning;
    }

    @FormElement(identifier = "pruning")
    public void setPruning(String pruning) {
        this.pruning = pruning;
    }

    public String getTrainingSize() {
        return trainingSize;
    }

    @FormElement(identifier = "trainingsize")
    public void setTrainingSize(String trainingSize) {
        this.trainingSize = trainingSize;
    }
}

```

Figure 4.38: The custom configuration object, using the *FormElement* annotations.

4.2.7.2 Implementation

After the configuration object has been created, the next step is to create a class named *CARTImpl* inside the package *at.tugraz.hidipic.web.execution.impl*, extending the *DefaultMethodImpl* class. This class has to override the *run()* method, where the execution of the algorithm will take place. The developer can freely create helper methods and such, but the *run()* method will be called during method execution. The parameters as well as the input files are inside the *Map < String, Object > data*, which is the method parameter of the *run()* method. Inside this data map, all relevant input objects can be found, identifiable by their names. The handling of these input objects will differ from algorithm to algorithm and has to be implemented according to the individual needs of the method at hand. If anything goes wrong, for example if an exception occurs, it is advised to use the *fireErrorEvent()* method, in order to let users know what went wrong, and that the method execution could not be performed successfully. Another method regarding events is *fireStartEvent*, which can be used, for example, after all input parameters and data have been validated and the execution can start. The third event function, *fireSuccessEvent*, is called automatically when the result creation is triggered.

Figure 4.39 shows a part of the *CARTImpl* class, as implemented within KN-ODWAT. Usually, after the validation of input data and necessary parameters, the algorithm execution can start, and again, if errors occur, an event should be fired. When the algorithm is completed, the created output data has to be saved and

to do so, there are two kinds of result classes available, the *ParamResult* and the *FileResult*. The *ParamResult* is basically just a String with a title, where the text and the numerical data can be stored, while the *FileResult* contains the link to files generated during the execution. This differentiation is relevant for the view later on and makes managing output results a lot easier. Once these result objects have been created, they are added to a list containing *GeneralResult* objects, describing a collection of output data. This output data list is then passed on to the method *createResult()*, which handles the persistence of a result object, adds and persists all the output data and fires the success event if everything worked, concluding the method execution. In essence, the developer responsible for adding a new method to the KNODWAT framework just has to create a class, override a method and within this method, create a list containing output data and call the *createResult()* method, which should be manageable for people with some experience in the Java programming language, guided by this tutorial.

```

@Component
public class CARTImpl extends DefaultMethodImpl {

    @Override
    public void run(Map<String, Object> data) {
        try {
            List<GenericResult> outputData = new ArrayList<GenericResult>();
            Map<String, String> parameters = (Map<String, String>) data.get("parameters");
            Map<String, String> inputDataStrings = (Map<String, String>) data.get("inputData");
            //get inputfiles
            boolean train = false;
            File trainFile = null;
            for (String s : inputDataStrings.keySet()) {
                String ss = inputDataStrings.get(s);
                if (s.equals("train")) {
                    train = true;
                    trainFile = getFile(ss);
                }
            }
            if (trainFile == null || !train) {
                fireErrorEvent("There was an error with the input file for CART.");
                return;
            }
        }
    }
}

```

Figure 4.39: Custom implementation class for the CART algorithm.

4.2.7.3 View and activation

Once both the configuration object and the implementation class have been created, positioned and named correctly, the only things left to do are to create a suitable view for the result detail and to create a database entry, which makes the method usable. The addition of the database entry is shown in figure 4.40. The method has to be named in the same way as the configuration object and the implementation class, and in the example of *CART*, the name has to be *CART*. This can be executed by an administrator using the administration panel.

The last step is to create a custom JSP view file for presenting the results generated by the newly added algorithm. To do this, a new JSP file named *CART.jsp* has to be

created, within the *customresult* folder, which can be found in *WEB-INF/jsp/*. The developer should have some basic knowledge on how to create Java Server Pages, especially if the goal is to present the results in an attractive and neat way, providing a high amount of transparency and clarity. Nonetheless, the creation of this view is, again, fairly simple. All of the output data objects are available in the $\{outputData\}$ object and accessible via the JSTL. The existing views for the CART and C4.5 implementations can be used as a template. There are no limits regarding frontend "magic" such as JavaScript based animations or widgets, as the created view will be included within the existing *result.jsp* view and is not restricted in any way.

After the view has been created, positioned, named correctly and the database entry has been added, the method can be used throughout the framework. It is of course advisable to do execute some intensive testing before releasing an extended version of the framework, as bugs and errors regarding the convention over configuration concept behind the extension feature can lead to instability throughout the whole application.

The screenshot shows a web-based administration interface. At the top, there is a navigation menu with links: Home, Upload, Create, Manage, Help, Logout. Below this, there is a secondary menu with links: Users, Projects, Subprojects, and Methods (which is highlighted in blue). The main content area is titled 'ADD METHOD' and contains a form with three input fields: 'Name' (containing 'CART'), 'Showname' (containing 'CART'), and 'Description' (containing 'This CART implementation (SimpleCART from WEKA), takes one ARFF File as'). Below the form are two buttons: 'Save' (blue) and 'Reset' (orange). On the right side of the page, there is a sidebar with 'admin Logout' and 'ACTIONS' (containing 'Settings' and 'Administration' buttons).

Figure 4.40: Adding the method to the database via the administration panel.

This page intentionally left blank

5. Results

This chapter presents a feature list of the completed KNODWAT framework, a setup guide as well as a small knowledge discovery study using two medical data sets, which was conducted using KNODWAT.

The KNODWAT (Knowledge Discovery With Advanced Techniques), an extensible application framework for testing knowledge discovery methods, in its current version, provides many features, with the most important ones being the following:

- Intuitive, web based user interface
- Localization for English and German
- High performance result creation
- Multi-file upload system
- Helpful documentation for beginners
- Event based notification system
- Dynamic content filtering methods for fast navigation
- Simple content management
- Easy extension capabilities for knowledge discovery algorithms
- Multiple user accounts and roles
- Full administration capabilities within the system
- Social features such as following, sharing and commenting
- Multi-platform compatibility

5.1 KNODWAT setup

5.1.1 Configuration

Especially for the purpose of multi-platform compatibility, there are some settings, which can be changed, if necessary, including file paths for the file upload functionality and the database server.

5.1.1.1 Database

In the file *application.properties*, within *src/main/resources*, the keys:

```
db.url = jdbc:mysql://localhost:3306/kdd
```

```
db.user = root
```

```
db.pass = root
```

can be changed if necessary, providing the location, database name, username and password of the MySQL server used for the application.

5.1.1.2 File Paths

In the file *application.properties*, within *src/main/resources*, the keys:

```
images.dynamic.path = file:/home/uploaddata/
```

```
upload.path = /home/uploaddata/
```

can be changed, to control where uploaded files are stored on the target system.

5.1.2 Building with Apache Maven

The KNODWAT source code can be built using Maven 2.2.1 or higher (available at <http://maven.apache.org/>), using the command *mvn clean package*, within the root folder (where the *pom.xml* file resides). A working Internet connection has to be available, in order for Maven to download all the dependency files. Once the building process has been executed successfully, a *.war* file is created in the *target* folder of the KNODWAT root folder, which can then be deployed on a web server.

5.1.3 Deployment on Apache Tomcat

In order to deploy the KNODWAT application, the *.war* file created during the building process has to be copied into the *webapps* folder of the installed Apache Tomcat instance. Newer Tomcat versions (the version used during development was 7.0), can be deployed on the fly, without the need of a restart. It is, however, advised to stop the server, copy the *.war* file into the *webapps* folder and then restart the server, to avoid complications. Once started, the server recognizes the *.war* file and deploys it automatically, if not configured otherwise. If this does not happen, the application has to be deployed manually, using the Apache Tomcat management interface. For more information and instructions regarding the Apache Tomcat server, please consult <http://tomcat.apache.org/>.

5.2 Benchmark

In order to test KNODWAT in regard to its usability, stability, usefulness and the correctness of the two implemented methods, a small study was performed. In this study, the two implemented algorithms, CART and C4.5 were tested on two different data sets provided by the UCI (Frank and Asuncion, 2010). The algorithms were trained using three different training set sizes, each with and without pruning, so all in all there were 6 classifiers trained per method and data set. The results of this study are presented in the following section.

5.2.1 Test data

The test data was acquired from the UCI - University of California, Irvine machine learning data sets (Frank and Asuncion, 2010). The two data sets used in this study were the Breast Cancer data set (Zwitter and Soklic, 1988) and (Frank and Asuncion, 2010) and the Hepatitis data set (Frank and Asuncion, 2010). Both have their origin in the medical domain. Example data rows for each set are:

- Breast Cancer :

'50-59', 'ge40', '15-19', '0-2', 'yes', '2', 'left', 'central', 'yes', 'no-recurrence-events'

'30-39', 'premeno', '30-34', '9-11', 'no', '2', 'right', 'left_u', 'yes', 'recurrence-events'

- Hepatitis

30, female, no, no, no, no, no, yes, no, no, no, no, no, 0.7, 100, 31, 4, 100, no, LIVE

59, female, no, no, yes, yes, no, yes, yes, yes, yes, no, no, 1.5, 107, 157, 3.6, 38, yes, DIE

More information on the different attributes and other interesting details regarding these data sets and many others can be found in (Frank and Asuncion, 2010).

In the following tables 5.1 and 5.2, the acronyms for the different test modi are listed and explained for further reference within the result images in 5.2.2.

Name	Data Set	Algorithm	Pruning	Training Set Size
BCC45Y30	Breast Cancer	C4.5	YES	30%
BCC45Y50	Breast Cancer	C4.5	YES	50%
BCC45Y70	Breast Cancer	C4.5	YES	70%
BCC45N30	Breast Cancer	C4.5	NO	30%
BCC45N50	Breast Cancer	C4.5	NO	50%
BCC45N70	Breast Cancer	C4.5	NO	70%
HEPC45Y30	Heptatitis	C4.5	YES	30%
HEPC45Y50	Heptatitis	C4.5	YES	50%
HEPC45Y70	Heptatitis	C4.5	YES	70%
HEPC45N30	Heptatitis	C4.5	NO	30%
HEPC45N50	Heptatitis	C4.5	NO	50%
HEPC45N70	Heptatitis	C4.5	NO	70%

Table 5.1: Table describing the different label codes for the C4.5 algorithm.

Name	Data Set	Algorithm	Pruning	Training Set Size
BCCARTY30	Breast Cancer	CART	YES	30%
BCCARTY50	Breast Cancer	CART	YES	50%
BCCARTY70	Breast Cancer	CART	YES	70%
BCCARTN30	Breast Cancer	CART	NO	30%
BCCARTN50	Breast Cancer	CART	NO	50%
BCCARTN70	Breast Cancer	CART	NO	70%
HEPCARTY30	Heptatitis	CART	YES	30%
HEPCARTY50	Heptatitis	CART	YES	50%
HEPCARTY70	Heptatitis	CART	YES	70%
HEPCARTN30	Heptatitis	CART	NO	30%
HEPCARTN50	Heptatitis	CART	NO	50%
HEPCARTN70	Heptatitis	CART	NO	70%

Table 5.2: Table describing the different label codes for the CART algorithm.

5.2.2 Test results

5.2.2.1 C4.5, Breast Cancer Results

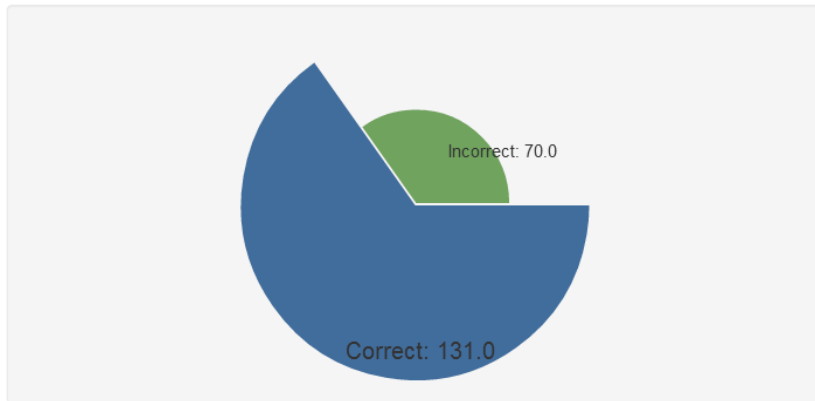


Figure 5.1: BCC45N30 Results

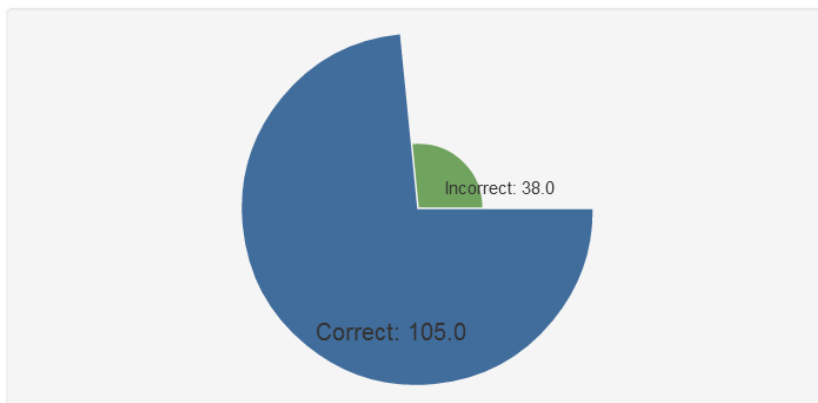


Figure 5.2: BCC45N50 Results

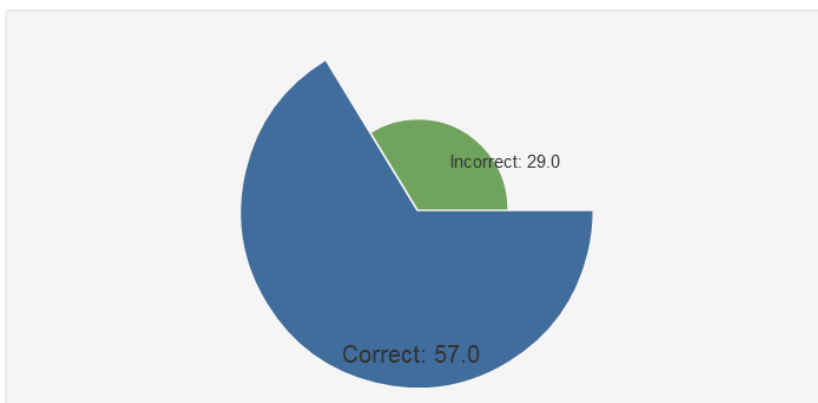


Figure 5.3: BCC45N70 Results

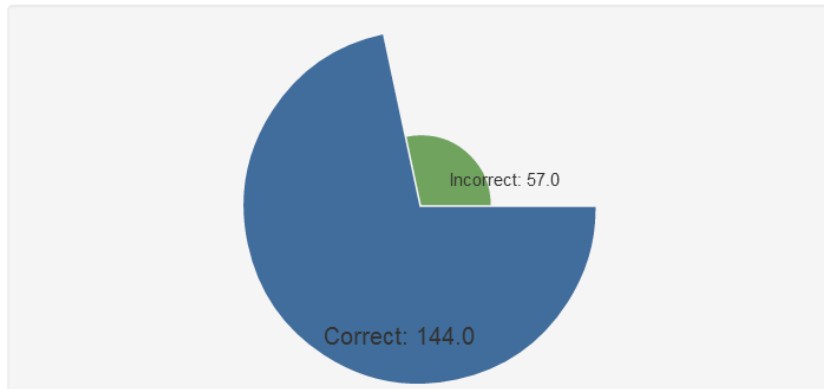


Figure 5.4: BCC45Y30 Results

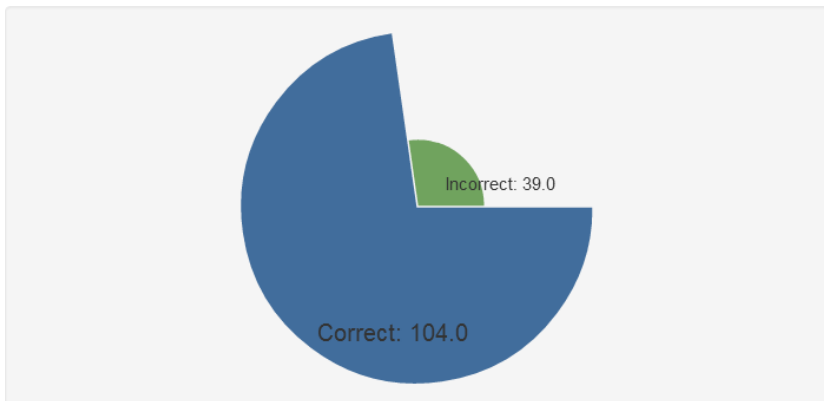


Figure 5.5: BCC45Y50 Results

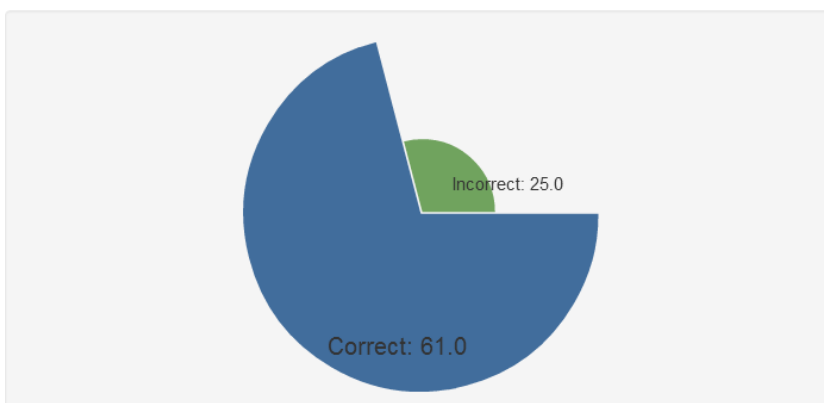


Figure 5.6: BCC45Y70 Results

5.2.2.2 C4.5, Hepatitis Results

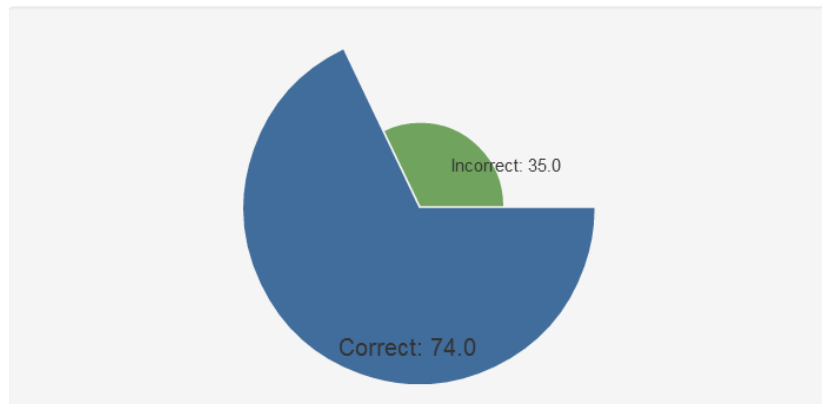


Figure 5.7: HEPC45N30 Results

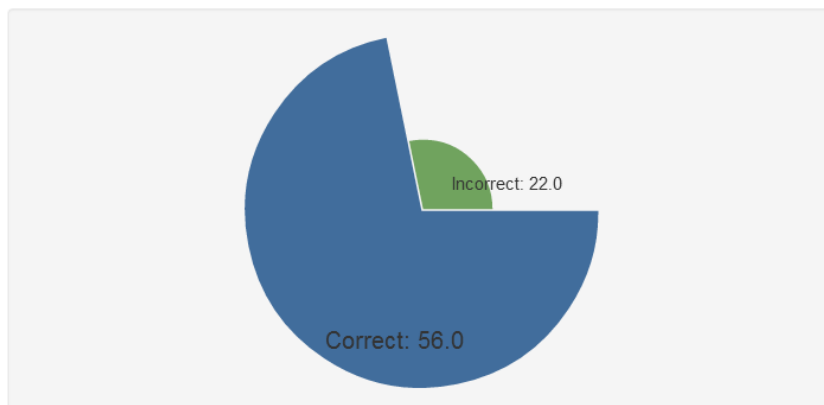


Figure 5.8: HEPC45N50 Results

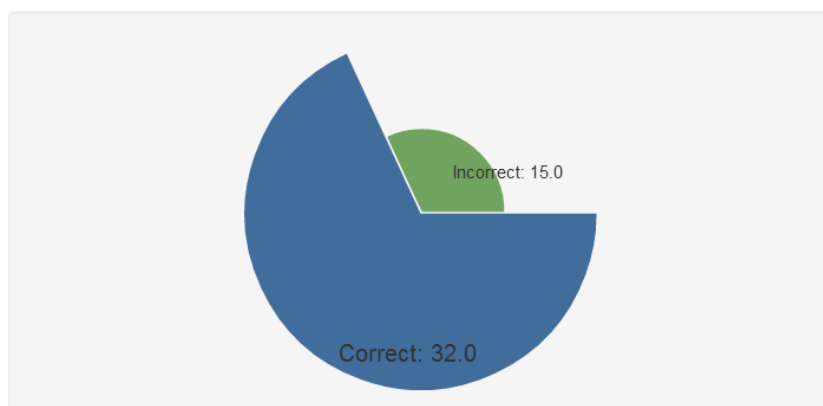


Figure 5.9: HEPC45N70 Results

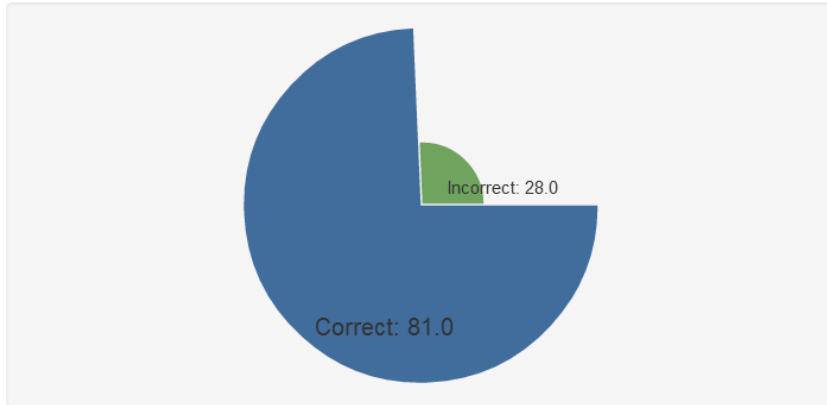


Figure 5.10: HEPC45Y30 Results

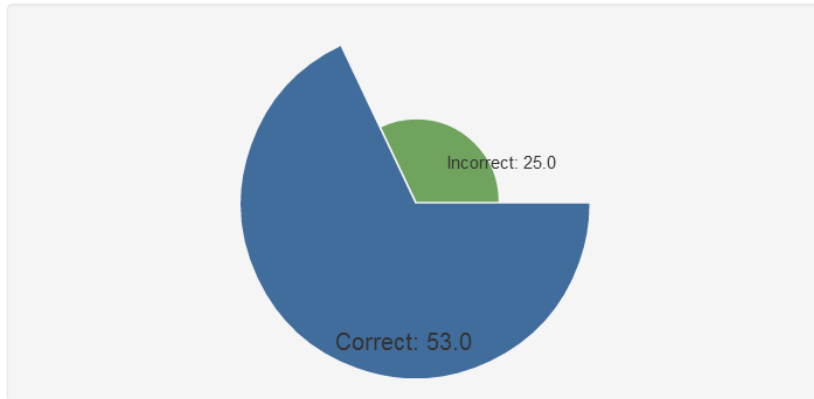


Figure 5.11: HEPC45Y50 Results

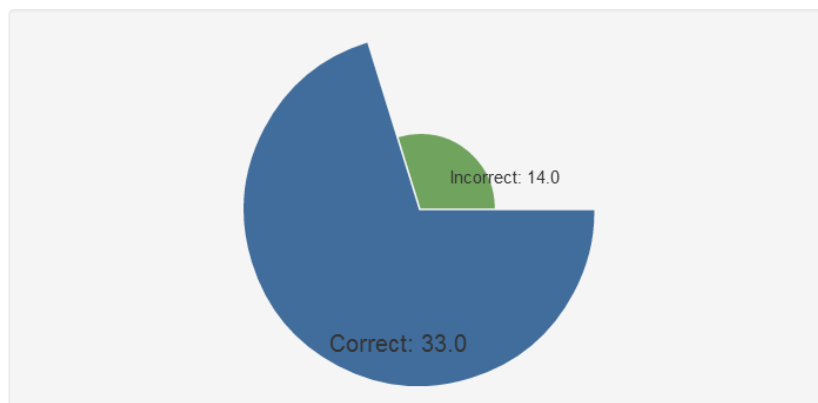


Figure 5.12: HEPC45Y70 Results

5.2.2.3 CART, Breast Cancer Results

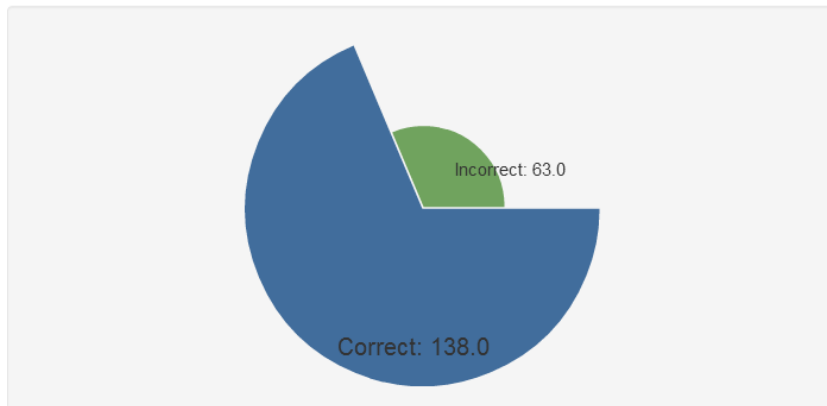


Figure 5.13: BCCARTN30 Results

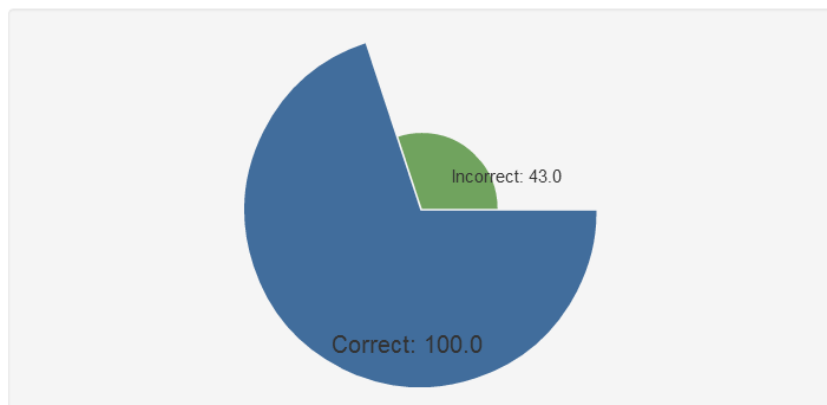


Figure 5.14: BCCARTN50 Results

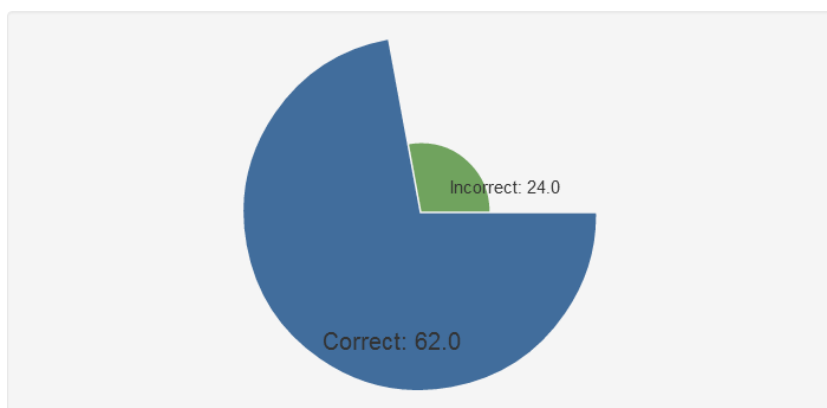


Figure 5.15: BCCARTN70 Results

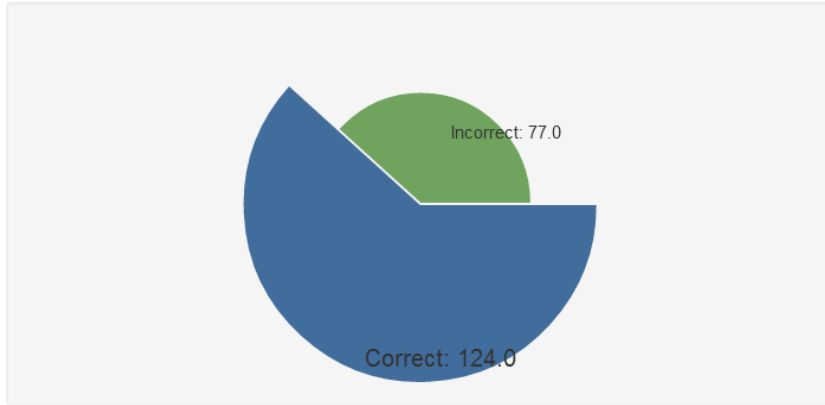


Figure 5.16: BCCARTY30 Results

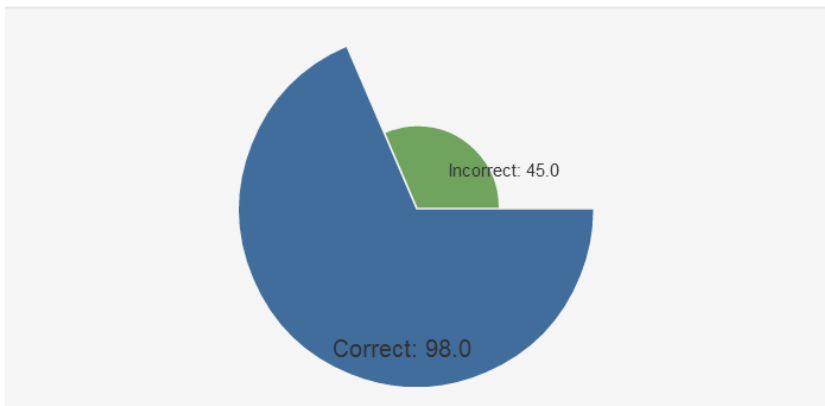


Figure 5.17: BCCARTY50 Results

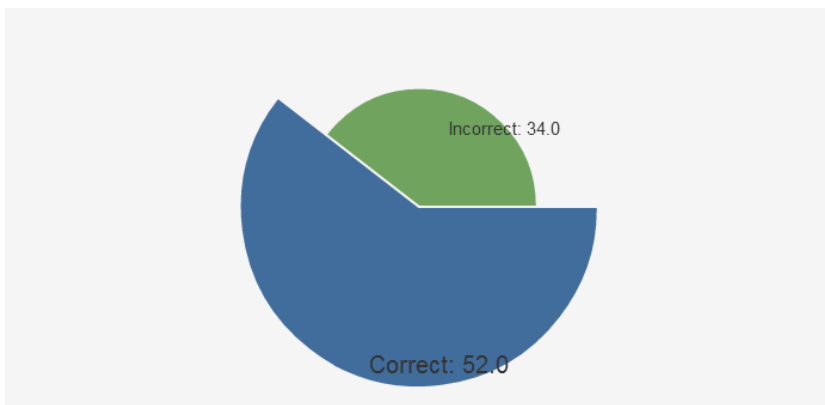


Figure 5.18: BCCARTY70 Results

5.2.2.4 CART, Hepatitis Results

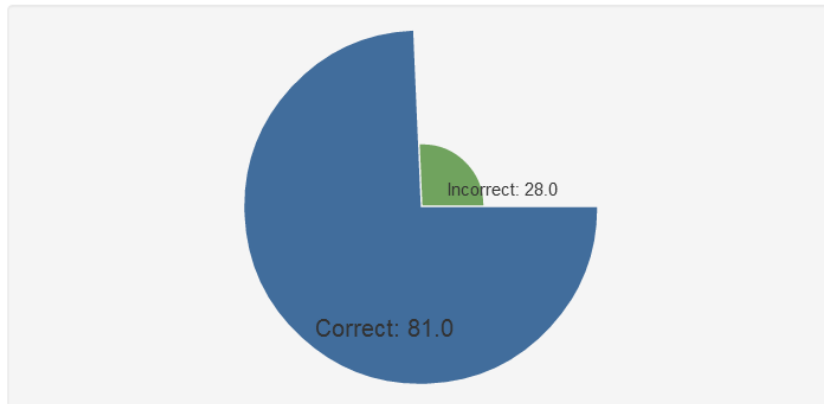


Figure 5.19: HEPCARTN30 Results

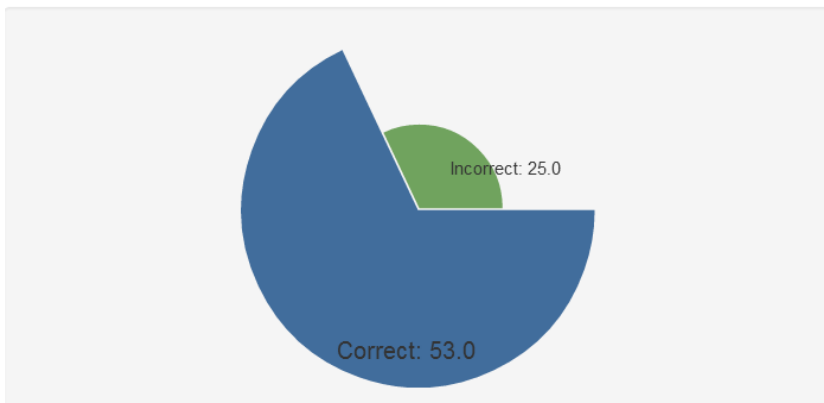


Figure 5.20: HEPCARTN30 Results

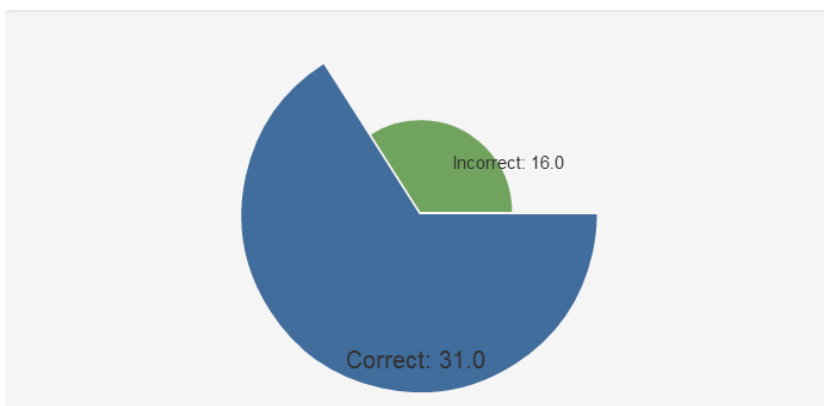


Figure 5.21: HEPCARTN30 Results

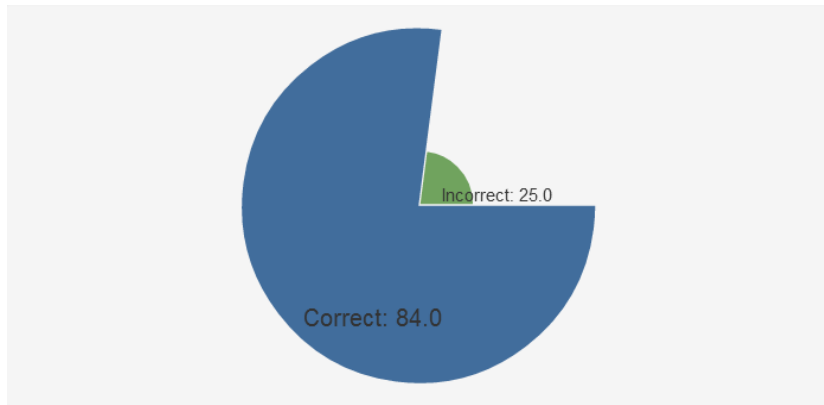


Figure 5.22: HEPARTY30 Results

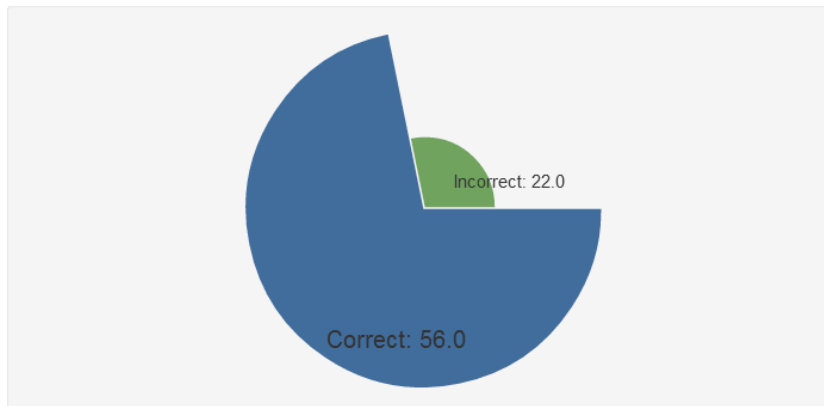


Figure 5.23: HEPARTY50 Results

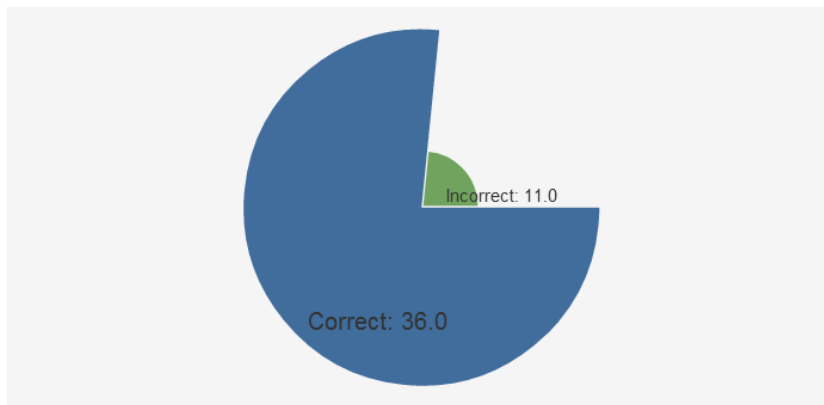


Figure 5.24: HEPARTY70 Results

5.2.3 Analysis

The result images in 5.2.2, were taken directly out of the KNODWAT result detail views. They represent a graph showing the fraction of correctly and incorrectly classified instances for the different classifiers. This section provides explanations and interpretations of the results generated throughout this small demonstrative study. First, the C4.5 results will be compared with each other on both test sets, then the CART results and, finally, the C4.5 results will be compared to the CART results in an attempt to find out, which algorithm had the better overall performance with respect to these two data sets.

5.2.3.1 C4.5

Figure 5.25 shows the six different configurations that were used during the C4.5 tests. The results are not very far apart, when there is tree pruning involved, without pruning however, the results differ greatly with a training size of 50% being the only result with comparable performance. In general, the two classifiers using a 50% training set size, both with and without pruning, performed best with a 73% chance to correctly classify instances from the test set.

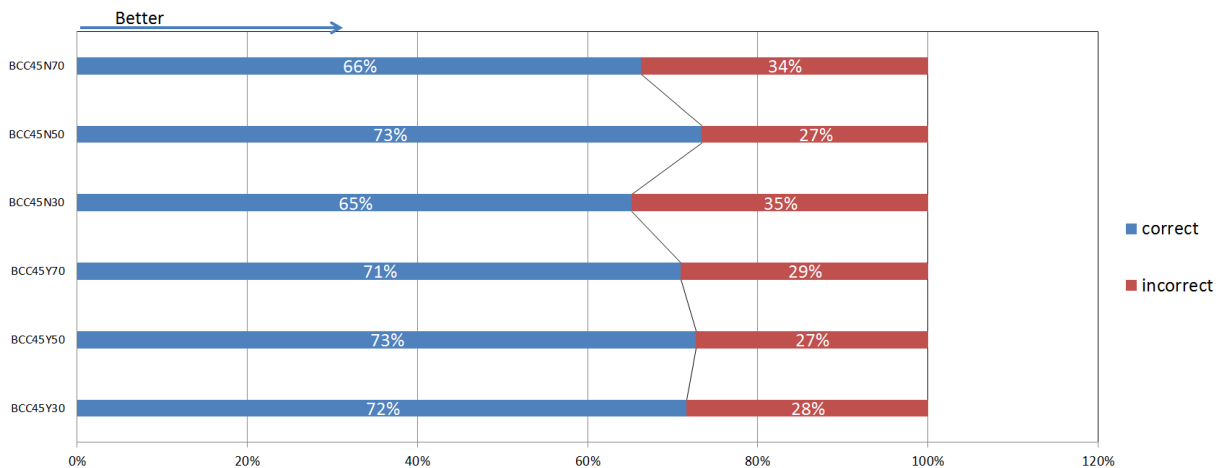


Figure 5.25: C4.5 Results on the Breast Cancer data set using different values for pruning and the size of the training set.

Figure 5.26 shows quite a different result on the Hepatitis data set. In this case, the difference between pruned and unpruned trees is fairly small on average, with the pruned trees still slightly outperforming the unpruned ones. More interestingly, however, the best overall result was achieved using a 30% training set size, obtaining a 74% correct classification rate on the test set. Within the unpruned classifiers, the 50% training set size variant performed best with 72% of correct classifications.

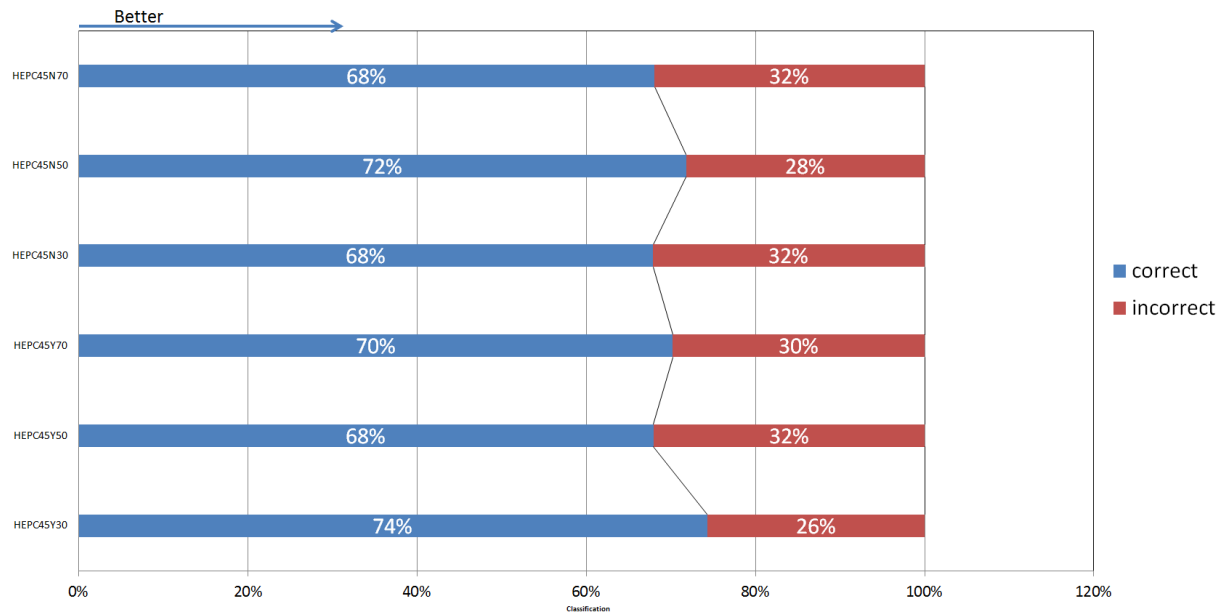


Figure 5.26: C4.5 Results on the Hepatitis data set using different values for pruning and the size of the training set.

5.2.3.2 CART

Figure 5.27 shows a very different picture than the C4.5 study on the Breast Cancer data set. The pruned classifiers performed a lot weaker than the unpruned ones, with the strongest one achieving 69% of correct classification using the 50% training set size variant. Within the unpruned classifiers, the results are very close together, with 72% for the 70% training set size outperforming the 70% correct classification rate of the 50% training set size result.

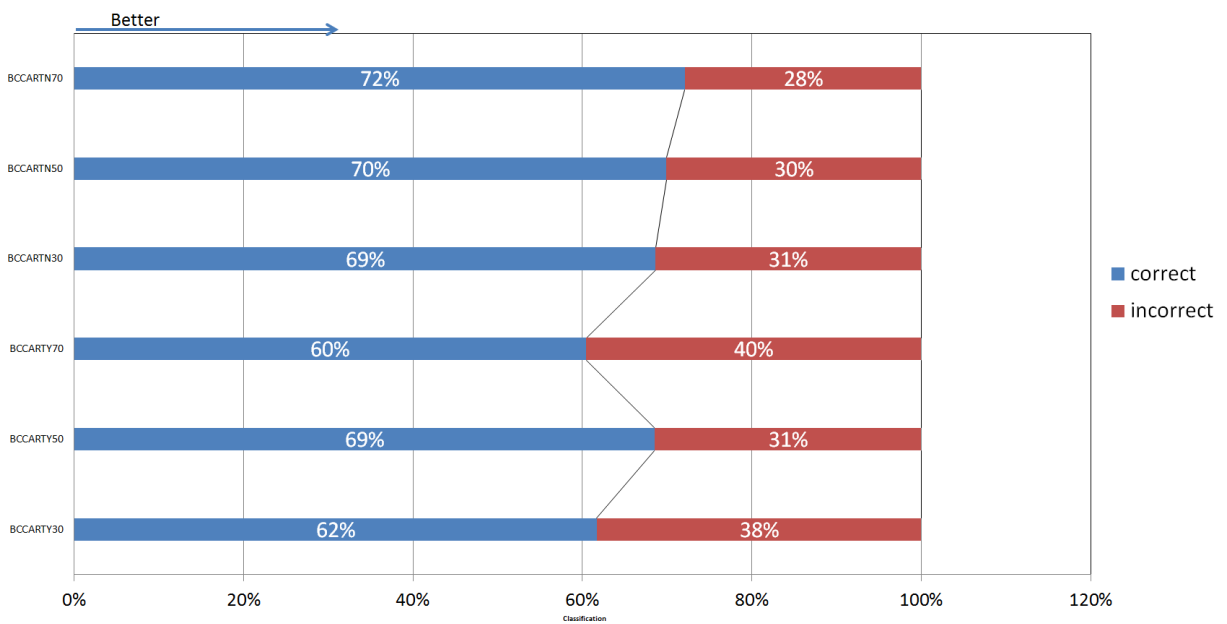


Figure 5.27: CART Results on the Breast Cancer data set using different values for pruning and the size of the training set.

Figure 5.28 shows the results of the different CART classifiers on the Hepatitis data set. It is noticeable, that the pruned trees performed way better than the unpruned trees, with the 30% and 70% training set sizes achieving a correct classification rate of 77% on their respective test sets. Within the unpruned classifiers, the 70% variant also performed well, with 74% correct classifications, especially compared to the other two variants, both scoring below 70% correct classifications.

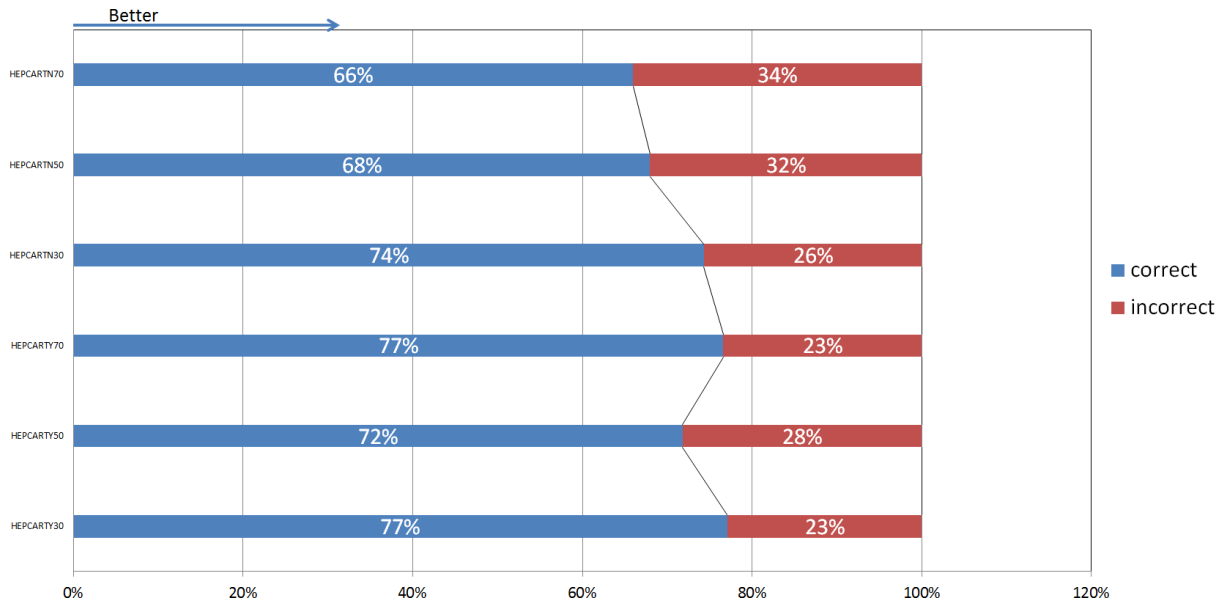


Figure 5.28: CART Results on the Hepatitis data set using different values for pruning and the size of the training set.

5.2.3.3 Comparison

Obviously, the two data sets, each containing merely 300 to 500 data sets and only 6 different versions of the two algorithms to be tested on the data are not highly representative in a context of actually gaining useful knowledge and insights from the data. This was, however, not the goal of the study in the first place, which was rather to evaluate the functionality and framework that the KNODWAT application provides to conduct such studies. Still, the results can be interpreted and compared, which can yield useful information on the usage and application of algorithms.

In figure 5.29, the two algorithms, CART and C4.5 and the results generated from their application to the Breast Cancer data set are displayed. The graph shows, that on the whole, the C4.5 algorithm outperformed CART with the used configurations, where only the CART variant without pruning and a training size of 70% comes close to the best result of C4.5, which is the no-pruning, 50% training set size result. It is however interesting to note, that the best variants for CART were among the worst ones for C4.5.

Figure 5.30 shows the comparison of CART and C4.5 on the Hepatitis data set, where the performance of the two algorithms is closer than on the Breast Cancer data set, and in this case, CART generated the best classifiers. The two best results of CART, the 77% correct classification rate of the pruned 30% and 70% training set sizes remained unbeaten by any of C4.5's results on this data set. For most of the configurations, the two algorithms behaved similarly, except in the case of the unpruned 50% training set size, where C4.5 created a fairly good classifier, while CART's result was a lot worse compared to that of the other configurations.

On the whole, the no-free-lunch theorem has been demonstrated in this small study as well, with each algorithm beating the other one on one of the data sets. Even in the case of these small data sets and a very limited range of different configurations using only two parameters, some fairly interesting results were generated by the use of the KNODWAT application. The program behaved as expected and made it very easy to conduct this study, with multi-file upload, different subprojects for the two studies and the intuitive user interface, with regard to the result creation and viewing being the most impacting factors throughout the experience.

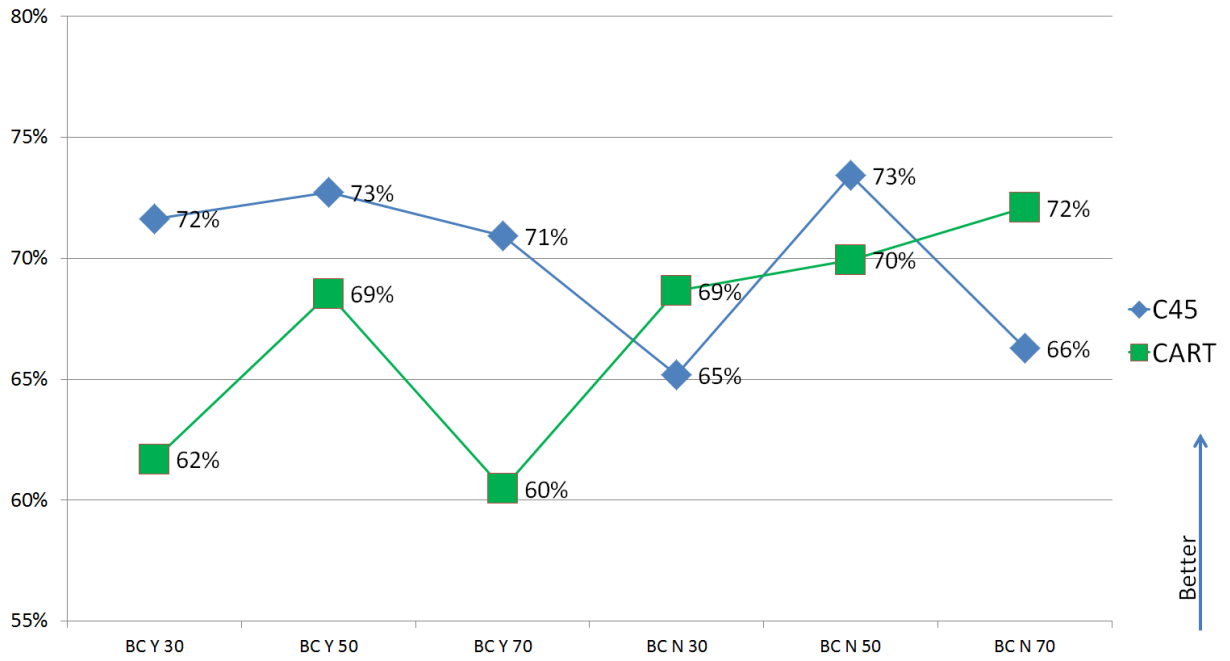


Figure 5.29: Comparison between CART and C4.5 on the Breast Cancer data set.

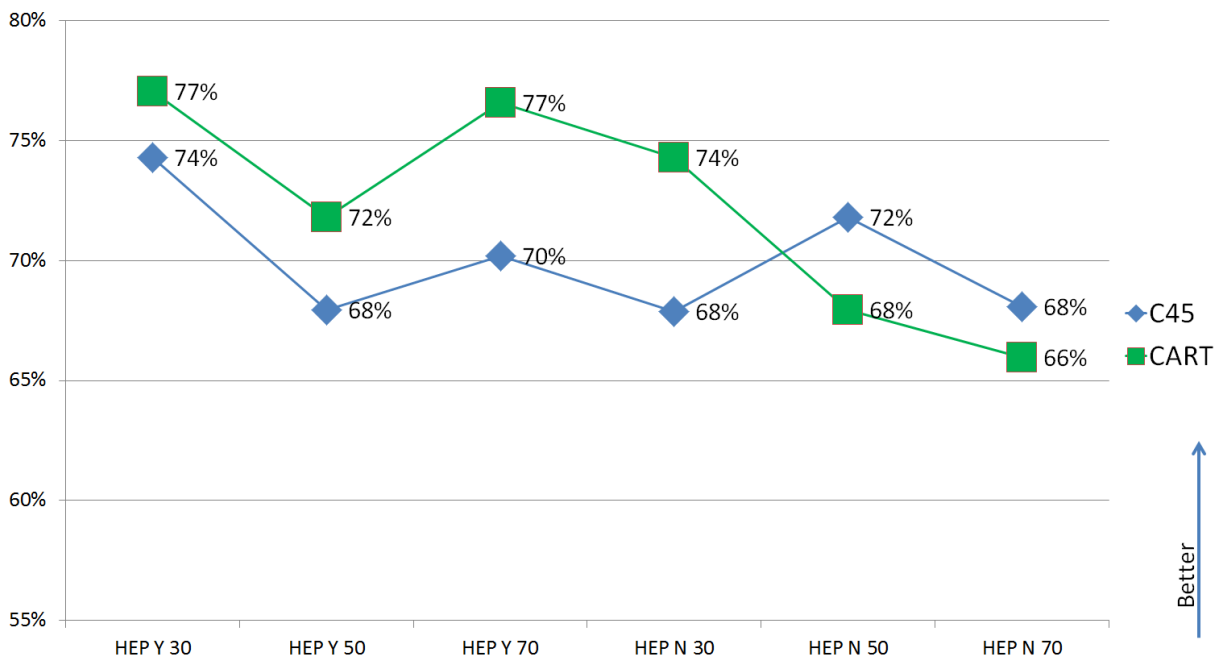


Figure 5.30: Comparison between CART and C4.5 on the Hepatitis data set.

This page intentionally left blank

6. Conclusions

This thesis introduced KNODWAT (Knowledge Discovery With Advanced Techniques), a framework for testing knowledge discovery methods with a focus on making it easy for developers to add new functionality to the system. The goal of this thesis was to provide a tool for both experts and beginners in the field of knowledge discovery, in order to enable researchers without extensive experience in data mining to use technologically advanced methods to uncover interesting patterns and relationships within their data. KNODWAT can be considered a web based application with a graphical user interface designed towards usability, if used correctly. Social features such as content sharing and the ability to express an opinion within the system as well as collaboration possibilities within projects, makes KNODWAT a modern environment for research groups. However, the application can not be extended without at least one expert who has programming experience and the skills to implement a certain knowledge discovery technique. The decision to create KNODWAT as a web based application has both advantages and disadvantages. On the one hand, many users will have an easier time getting started with a web based system due to the experiences they made with other systems of the kind, such as social networks or other prominent web sites. Web based applications also have the advantage of being very connectable to external services and inherently create connections between users and their generated content. On the other hand, however, there may be limitations considering methods with high computational complexity or very specific and expensive graphical representations of results, as they can be harder to implement in a web based application, than in a native client. Nonetheless, with the trend of mobile devices becoming more and more capable and providing improved user interaction features, it was very important to make KNODWAT available for as many platforms as possible, which is definitely a strength of applications

developed for the web.

On the whole, the idea of a globally connected research platform, making knowledge and the methods used to acquire it available to everyone, is very intriguing and KN-ODWAT is a small step towards that direction.

Conducting a survey and writing a paper with the goal of bringing popular knowledge discovery techniques to a wider audience is no easy task. It is often hard to find a good balance between technical details and a comprehensible explanation. Another problem is the choice of references, deciding where to lead a reader who is interested in learning more about the topic. It is plausible, that a basic explanation of a highly complex method can awaken people's interest, which is then utterly demolished, if they want to delve deeper and find themselves overwhelmed with content which was written for readers with a higher amount of previous knowledge on the subject. Because of this problem, the process of leading a wider audience to complex methods of knowledge discovery has to happen in multiple steps, starting with an easy to read summary of the basic functionality, application areas, advantages, disadvantages and available software of a method, which is what was attempted during this thesis. The next step would be to lead the reader from this basic summary to a more technical, but not overwhelmingly complex explanation of the algorithm's inner functionalities, which could further lead to actual, academic level work. This way, a natural progression takes place for researchers who are interested, but inexperienced.

The goal, to make more researchers aware of the many different, well studied and tested, approaches available to analyze their data, can not be achieved with a single article, thesis or book. Every discipline of science can benefit greatly from interdisciplinary work, where people work together and provide new methods and techniques for solving problems. However, in order to make these methods sustainable throughout the real world, there has to be an easily accessible documentation as well as a software designed on the principles of usability and human computer interaction (Holzinger, 2012b).

7. Future Work

Due to the modular and extensible concept behind object oriented software engineering, the KNODWAT framework can never be finished as such, meaning that there is always room for improvement within the existing functionality as well as some potential to further increase the number of available features and the quality of the software itself. This chapter presents a few points of direction which could be followed in order to improve KNODWAT. Some of the following ideas would require a fairly small amount of time to implement, but did not make it into the current release due to time constraints, others have the potential to be the basis of a whole thesis.

Knowledge discovery methods

With KNODWAT being a framework for testing different knowledge discovery methods, there is an essential need for many different algorithms to be available. During the course of this thesis, only two general methods could be implemented, in order to demonstrate the functionality of the software, but due to the easily extensible nature of the framework and the availability of sources like WEKA for algorithm implementations, it should not be a big problem to significantly increase the library of available methods.

Multi-method feature

In order to test and compare different methods, it would be both time efficient and intuitive, to use a single data set and test it with multiple methods at once. The result of such an execution would just be a performance chart of the different methods. This feature would describe the problem of finding the most suitable

method for the data set better than the current functionality, where every method has to be tested independently and then compared to the result of the others.

Usability testing

A formal usability test based on traditional usability inspection methods (Holzinger, 2005) or Collaborative approaches (Lockwood, 2003). The test subjects should probably be people with a background in scientific research applications and with limited experience in IT.

Testing framework

Unfortunately, building a stable test environment with a high coverage of unit tests was not possible within the given time, but would be a great addition to the framework. While Spring and Hibernate based functionality is less prone to errors, other types of functionality, especially concerning the extensibility features and the implemented algorithms, would definitely be improved by a test suite. Such a framework, containing both integration tests as well as unit tests could also be helpful for new developers trying to extend the framework, to make sure that the system still works after the addition of new methods.

Script language extensibility

A concept used by the Orange data mining software, is to provide the users with the ability to write frontend and backend addons using a script language such as python. This could make the extension of the KNODWAT framework even more flexible, especially in regard to customizing the result creation view.

Data deletion

In the current version of KNODWAT, data deletion has been deactivated. This was a conscious decision, based on the need to test the platform with a large amount of data as well as the general need to guarantee data consistency for scientific research. The implemented approach, which has been deactivated, lets users delete only con-

tent with no other data rows depending on it, which is fine, but not very intuitive to use, as a user would have to delete every single element manually, before a highly interconnected component could be deleted.

Another approach would be, to introduce a "deleted" flag, making the records invisible to the user, but still available and recoverable in the database.

Performance

The overall performance of the framework is good for the amounts of data, with which it was tested. However, with a large amount of users creating large amounts of data, this can quickly change. This means that, standard performance increasing mechanisms such as caching of static content, query indexing and other more problem specific methods should be implemented if necessary.

Security

The KNODWAT framework provides a secure way to log in and, by enforcing user restrictions, to hide data from other researchers. The security within the system, however, was not a point of great focus during development, but basic attacks such as sql-injection or cross-side-scripting are avoided by the applied technology by default. If the application is however to be used within a security sensitive context, for example using data that should stay private at all costs, a security audit would be advisable, as well as the use of Secure Socket Layer (SSL) technology and another authentication layer on the server's file system.

Localization

At the moment, the KNODWAT framework supports two languages, English and German. This is due to the fact, that these two languages are the only ones the developer felt comfortable doing translations in. If, however, the application is to be used for a highly international audience, additional languages should be added.

Multi platform testing

While Spring framework and Hibernate, when used as a basis for the application already provide a lot of multi platform compatibility, it was not feasible within the given time frame, to test KNODWAT on a lot of different platforms and with different database servers. In it's current version, the software has been successfully tested in all modern browsers (Chrome 21, Firefox 12, Opera 12) except Internet Explorer. It has also been successfully deployed and tested on an Apache Tomcat 7.0 on both a Windows 7 and a Linux Mint 13 machine, using MySQL Server 5.5.

List of Figures

1.1	The Knowledge Discovery Process (Fayyad et al., 1996)	16
2.1	The Process of Machine Learning (Kotsiantis, 2007) Taken from the slides of Holzinger (2012a)	20
2.2	A simple Decision Tree created in Matlab using CART	22
2.3	Optimal hyperplane for a linearly separable case (Wu and Kumar, 2009)	30
2.4	Mapping the training data to a higher dimension feature space ($\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$), constructing a separating hyperplane (Scholkopf and Smola, 2001)	32
2.5	Simple example of a Bayesian network representing the impact of different variables on each other. Every variable has a probability table describing its conditional distribution regarding its parents. (Nadkarni and Shenoy, 2001)	36
2.6	Dynamic Bayesian network representing temporal dependency between Y_1, Y_2 and Y_3 (Maimon and Rokach, 2010)	39
2.7	Fitting of a regression model using MATLAB's polyfit function on a demo dataset.	42
2.8	Regression tree describing predicted mileage of cars. Image source: http://www.statmethods.net/advstats/cart.html	44
2.9	Iterations of the k-means algorithm on a simulated dataset by (Hastie et al., 2008)	48
2.10	Network visualization of knowledge relationships between scientists in a study performed by IBM (Cross et al., 2002)	55

2.11	Social Network representing faculty members and their areas of expertise within a department of biomedical informatics by Merrill and Hripesak (2008)	56
2.12	Visualization of the top ranked image search results for the keyword "Mona Lisa" using PageRank by (Jing and Baluja, 2008)	58
2.13	Example visualization of a suspicious fraudulent network using the SAS Fraud Framework for Insurance, Source: http://www.sas.com	60
2.14	Number of interesting association rules from (Kianmehr et al., 2011), with different minimum-support values	62
2.15	Pseudo code of Apriori's association rule generation by Wu and Kumar (2009)	64
2.16	Visualization of the solution of a parameter-less evolutionary algorithm for the problem of network expansion by Lobo and Goldberg (2004)	72
2.17	Representation of polygonal approximation using genetic algorithms by Yin (1998)	74
2.18	Example program in tree form in a genetic programming algorithm (Tsakonas et al., 2004)	75
2.19	Visualization of a single layer feed forward neural network (Branch et al., 2008)	80
2.20	A three unit Hopfield Network (a) and adding three units to it (b) (Bersini and Sener, 2002)	83
2.21	Visualization of a 15x15 Kohonen Net (Wyns et al., 2004)	84
2.22	Example visualization of Particle Swarm Optimization in the context of clustering (Chang et al., 2012)	87
4.1	Screenshot of the "Manage Projects"-Area of the KNODWAT application.	101
4.2	The entity class for the comments-table, showing the annotations used for the object relational mapping.	107
4.3	Entity-relationship diagram, generated from the live database of the KNODWAT application.	108

4.4	The basis for the Object Relational Mapping from which the data model is created.	110
4.5	DataAccessService with the different controller classes using it.	111
4.6	DataAccessService with the different controller classes using it.	113
4.7	Class diagram of the method execution module	115
4.8	Structure of JSP-files (views) within the KNODWAT application. . .	117
4.9	Form for editing an existing project	118
4.10	Multiselect widget in the forms of KNODWAT	118
4.11	Table filtering via JavaScript.	119
4.12	List filtering via JavaScript.	119
4.13	Landing page without following subprojects, containing only statistics about the database.	120
4.14	Landing page when following subprojects, with an overview of the projects and recent events within them.	120
4.15	The Help page, to introduce new users to the system.	121
4.16	Default error page.	121
4.17	Overview of the subprojects within the administration panel	123
4.18	Editing a user.	123
4.19	Creating and editing a project inside the admin panel.	123
4.20	Selection of the project, within the result will be created.	124
4.21	Selection of a subproject and the method for the result.	124
4.22	Specifying parameters and input data for the result creation.	125
4.23	Project management overview part 1	126
4.24	Project management overview part 2	127
4.25	Uploading a file, to create input data.	128
4.26	Settings page with follow and unfollow options.	129
4.27	Commenting on results, projects and subprojects is possible within the KNODWAT application.	130
4.28	Overview of recent events happening in related projects.	131

4.29	Configuration object annotated using the <i>FormElement</i> annotation.	132
4.30	Custom parameters displayed for every different method.	133
4.31	German localization is supported within the KNODWAT application.	135
4.32	Basic form validation where no special characters and length limits are in place.	136
4.33	Spring Security config, showing user roles and their restrictions.	137
4.34	The login screen.	138
4.35	If the user does not have sufficient rights to view a site, this error page is displayed.	138
4.36	Result detail part 1	139
4.37	Result detail part 2	140
4.38	The custom configuration object, using the <i>FormElement</i> annotations.	143
4.39	Custom implementation class for the CART algorithm.	144
4.40	Adding the method to the database via the administration panel.	145
5.1	BCC45N30 Results	152
5.2	BCC45N50 Results	152
5.3	BCC45N70 Results	152
5.4	BCC45Y30 Results	153
5.5	BCC45Y50 Results	153
5.6	BCC45Y70 Results	153
5.7	HEPC45N30 Results	154
5.8	HEPC45N50 Results	154
5.9	HEPC45N70 Results	154
5.10	HEPC45Y30 Results	155
5.11	HEPC45Y50 Results	155
5.12	HEPC45Y70 Results	155
5.13	BCCARTN30 Results	156
5.14	BCCARTN50 Results	156
5.15	BCCARTN70 Results	156

5.16	BCCARTY30 Results	157
5.17	BCCARTY50 Results	157
5.18	BCCARTY70 Results	157
5.19	HEPCARTN30 Results	158
5.20	HEPCARTN30 Results	158
5.21	HEPCARTN30 Results	158
5.22	HEPCARTY30 Results	159
5.23	HEPCARTY50 Results	159
5.24	HEPCARTY70 Results	159
5.25	C4.5 Results on the Breast Cancer data set using different values for pruning and the size of the training set.	160
5.26	C4.5 Results on the Hepatitis data set using different values for pruning and the size of the training set.	161
5.27	CART Results on the Breast Cancer data set using different values for pruning and the size of the training set.	162
5.28	CART Results on the Hepatitis data set using different values for pruning and the size of the training set.	163
5.29	Comparison between CART and C4.5 on the Breast Cancer data set.	165
5.30	Comparison between CART and C4.5 on the Hepatitis data set.	165

This page intentionally left blank

List of Tables

- 5.1 Table describing the different label codes for the C4.5 algorithm. . . . 151
- 5.2 Table describing the different label codes for the CART algorithm. . . 151

This page intentionally left blank

References

- Agrawal, Rakesh, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo [1996]. *Fast discovery of association rules*. pages 307–328.
- Agrawal, Rakesh and Ramakrishnan Srikant [1994]. *Fast Algorithms for Mining Association Rules*. *Organization*, 1215(1558601538), pages 487–499. ISSN 15426270. doi:10.1.1.40.6757.
- Al-Sultana, Khaled S. and M. Maroof Khan [1996]. *Computational experience on four algorithms for the hard clustering problem*. *Pattern Recognition Letters*, 17(3), pages 295–308. ISSN 01678655. doi:10.1016/0167-8655(95)00122-0.
- Almuallim, Hussein [1996]. *An efficient algorithm for optimal pruning of decision trees*. *Artificial Intelligence*, 83(2), pages 347–362. ISSN 00043702. doi:10.1016/0004-3702(95)00060-7.
- Alsabti, Khaled, Sanjay Ranka, and Vineet Singh [1998]. *CLOUDS: A Decision Tree Classifier for Large Datasets*.
- Aumann, Yonatan and Yehuda Lindell [1999]. *A statistical theory for quantitative association rules*. *KDD 99 Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 20(3), pages 261–270. doi:10.1145/312129.312243.
- Back, Thomas [2000]. *Evolutionary Computation 2 Advanced Algorithms and Operators*. Number v. 1 in Evolutionary computation, Institute of Physics Publishing. ISBN 0750306653, 25–37 pages.
- Baker, E. and A. K. Jain [1976]. *On feature ordering in practice and some finite*

- sample effects. In *In Proceedings of the Third International Joint Conference on Pattern Recognition*, pages 45–49.
- Banfield, J D and A E Raftery [1993]. *Model-based Gaussian and non-Gaussian clustering*. *Biometrics*, 49(3), pages 803–821. ISSN 0006341X. doi:10.2307/2532201.
- Bastide, Yves, Rafik Taouil, Nicolas Pasquier, Gerd Stumme, and Lotfi Lakhal [2000]. *Mining Frequent Patterns with Counting Inference*. *ACM SIGKDD Explorations*, 2(2), pages 66–75.
- Batista, Gustavo E. A. P. A. and Maria Carolina Monard [2003]. *An Analysis of Four Missing Data Treatment Methods for Supervised Learning*. *Applied Artificial Intelligence*, pages 519–533.
- Bayardo Jr., Roberto J Bayardo [1998]. *Efficiently Mining Long Patterns from Databases*. *SIGMOD 98 Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 85–93. doi:10.1145/276304.276313.
- Ben-Bassat, M. [1978]. *Myopic Policies in Sequential Classification*. *IEEE Transactions on Computers*, C-27(2), pages 170–174. ISSN 0018-9340. doi:10.1109/TC.1978.1675054.
- Beni, G and J Wang [1989]. *Swarm intelligence in cellular robotic systems*.
- Bennett, Kristin P., Kristin P Bennett, and O. L Mangasarian [1992]. *Multicategory Discrimination via Linear Programming*. *OPTIMIZATION METHODS AND SOFTWARE*, 3, pages 27 – 39.
- Bentley, Jon Louis and Jerome H Friedman [1978]. *Fast Algorithms for Constructing Minimal Spanning Trees in Coordinate Spaces*. *IEEE Trans on Computers TOC*, C-27(2), pages 97–105.
- Bersini, Hugues and Pierre Sener [2002]. *The connections between the frustrated chaos and the intermittency chaos in small Hopfield networks*. *Neural Networks*, 15(10), pages 1197–1204. ISSN 08936080. doi:10.1016/S0893-6080(02)00096-5.
- Berthold, Michael R, Nicolas Cebron, Fabian Dill, Thomas R Gabriel, Tobias Kötter, Thorsten Meinl, Peter Ohl, Christoph Sieb, Kilian Thiel, and Bernd

- Wiswedel [2008]. *KNIME: The Konstanz Information Miner. Data Analysis Machine Learning and Applications*, 11(1), pages 319–326. ISSN 14318814. doi:10.1007/978-3-540-78246-9.
- Bohanec, Marko and Ivan Bratko [1994]. *Trading Accuracy for Simplicity in Decision Trees. Machine Learning*, 15(3), pages 223–250. ISSN 0885-6125. doi:10.1023/A:1022685808937.
- Bonchi, Francesco and Claudio Lucchese [2004]. *On Closed Constrained Frequent Pattern Mining*. pages 35–42.
- Boulicaut, Jean-François, Artur Bykowski, and Christophe Rigotti [2003]. *Free-Sets: A Condensed Representation of Boolean Data for the Approximation of Frequency Queries. Data Mining and Knowledge Discovery*, 7(1), pages 5–22. ISSN 1384-5810. doi:10.1023/A:1021571501451.
- Boulicaut, Jean-François and Baptiste Jeudy [2000]. *Using Constraints During Set Mining: Should We Prune or not?*
- Boulicaut, Jean-François, Artur Bykowski, and Christophe Rigotti [2000]. *Approximation of Frequency Queries by Means of Free-Sets*. pages 75–85.
- Branch, Ben, Jia Wang, and Taewon Yang [2008]. *A note on takeover success prediction. International Review of Financial Analysis*, 17(5), pages 1186–1193. ISSN 10575219. doi:10.1016/j.irfa.2007.07.003.
- Breiman, L, J H Friedman, R A Olshen, and C J Stone [1984]. *Classification and Regression Trees, Statistics/Probability Series*, volume 19. Wadsworth. ISBN 0412048418, 368 pages.
- Breiman, Leo [1996]. *Bagging predictors. Machine Learning*, 24(2), pages 123–140. ISSN 0885-6125. doi:10.1007/BF00058655.
- Breiman, Leo [2001]. *Random Forests. Machine Learning*, 45(1), pages 5–32. ISSN 0885-6125. doi:10.1023/A:1010933404324.
- Brin, Sergey, Rajeev Motwani, Jeffrey D Ullman, and Shalom Tsur [1997]. *Dynamic itemset counting and implication rules for market basket data. Proceedings of the*

- 1997 ACM SIGMOD international conference on Management of data SIGMOD 97, 26(2), pages 255–264. ISSN 01635808. doi:10.1145/253260.253325.
- Bull, L and T Kovacs [2010]. *Foundations of Learning Classifier Systems*. Studies in Fuzziness and Soft Computing, Springer. ISBN 9783642064135.
- Bullnheimer, Bernd, Richard F Hartl, and Christine Strauss [1997]. *A New Rank Based Version of the Ant System — A Computational Study*. *Central European Journal for Operations Research and Economics*, 7, pages 25–38. doi:10.1.1.49.4735.
- Buntine, Wray and Tim Niblett [1992]. *A Further Comparison of Splitting Rules for Decision-Tree Induction*. *Machine Learning*, 8(1), pages 75–85. ISSN 0885-6125. doi:10.1023/A:1022686419106.
- Calders, Toon and Bart Goethals [2002]. *Mining All Non-Derivable Frequent Itemsets*. *Principles of Data Mining and Knowledge Discovery*, 2431, pages 74–85.
- Can, Fazli [1993]. *Incremental clustering for dynamic information processing*. *ACM Transactions on Information Systems*, 11(2), pages 143–164. ISSN 10468188. doi:10.1145/130226.134466.
- Carvalho, Deborah R. and Alex A. Freitas [2004]. *A hybrid decision tree/genetic algorithm method for data mining*. *Information Sciences*, 163(1-3), pages 13–35. ISSN 00200255. doi:10.1016/j.ins.2003.03.013.
- Cerrito, Patricia B [2006]. *Introduction to data mining using SAS Enterprise Miner Development*.
- Chakrabarti, Soumen, Byron Dom, Rakesh Agrawal, and Prabhakar Raghavan [1998]. *Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies*. *The VLDB Journal The International Journal on Very Large Data Bases*, 7(3), pages 163–178. ISSN 10668888. doi:10.1007/s007780050061.
- Chang, Pei-Chann, Jyun-Jie Lin, and Chen-Hao Liu [2012]. *An attribute weight assignment and particle swarm optimization algorithm for medical database classifications*. *Computer methods and programs in biomedicine*, 107(3), pages 382–92. ISSN 1872-7565. doi:10.1016/j.cmpb.2010.12.004.

- Changchien, S Wesley and Tzu-Chuen Lu [2001]. *Mining association rules procedure to support on-line recommendation by customers and products fragmentation*. *Expert Systems with Applications*, 20(4), pages 325–335. ISSN 09574174. doi:10.1016/S0957-4174(01)00017-3.
- Charniak, Eugene [1991]. *Bayesian Networks Without Tears*. *AI MAGAZINE*, 12(4), pages 50 – 63.
- Cheeseman, Peter and John Stutz [1996]. *Bayesian Classification(AutoClass):Theory and Results*.
- Chen, Pai-hsuen, Chih-jen Lin, and Bernhard Schölkopf [2005]. *A tutorial on ν -Support Vector Machines*.
- Chen, Xiaojun, Yunming Ye, Graham Williams, and Xiaofei Xu [2007]. *A Survey of Open Source Data Mining Systems*. *Science And Technology*, 4819(60603066), pages 3–14. doi:10.1007/978-3-540-77018-3_2.
- Cheng, B and M Titterington [1994]. *Neural networks: a review from a statistical perspective (with discussion)*. *Statistical Science*, 9, pages 2–54.
- Cheng, Jian and Marek J Druzdzel [2000]. *AIS-BN: An Adaptive Importance Sampling Algorithm for Evidential Reasoning in Large Bayesian Networks*. *JOURNAL OF ARTIFICIAL INTELLIGENCE RESEARCH*, 13, pages 13 – 155.
- Cleveland, William S. [1979]. *Robust Locally Weighted Regression and Smoothing Scatterplots*. *Journal of the American Statistical Association*, 74(368), page 829. ISSN 01621459. doi:10.2307/2286407.
- Cohen, E, M Datar, S Fujiwara, A Gionis, P Indyk, R Motwani, J D Ullman, and C Yang [2000]. *Finding interesting associations without support pruning*. doi:10.1109/69.908981.
- Cooper, Gregory F. [1990]. *The computational complexity of probabilistic inference using bayesian belief networks*. *Artificial Intelligence*, 42(2-3), pages 393–405. ISSN 00043702. doi:10.1016/0004-3702(90)90060-D.

- Cooper, Gregory F. and Edward Herskovits [1992]. *A Bayesian Method for the Induction of Probabilistic Networks from Data*. *Machine Learning*, 9(4), pages 309–347. ISSN 0885-6125. doi:10.1023/A:1022649401552.
- Cortes, Corinna and Vladimir Vapnik [1995]. *Support-Vector Networks*. *Machine Learning*, 20(3), pages 273–297. ISSN 0885-6125. doi:10.1023/A:1022627411411.
- Couzin, Iain D, Jens Krause, Richard James, Graeme D Ruxton, and Nigel R Franks [2002]. *Collective memory and spatial sorting in animal groups*. *Journal of Theoretical Biology*, 218(1), pages 1–11.
- Cross, Rob, Andrew Parker, and Stephen Borgatti [2002]. *A bird’s-eye view: Using social network analysis to improve knowledge creation and sharing*.
- Daniel Fu, Emilio Remolina [2003]. *A CBR Approach to Asymmetric Plan Detection*. In *Proceedings of Workshop on Link Analysis for Detecting Complex Behavior*.
- Darwin, C [1872]. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. sixth Edition. J. Murray. ISBN 0486450066.
- Deb, Kalyanmoy and David E. Goldberg [1989]. *An investigation of niche and species formation in genetic function optimization*. pages 42–50.
- Deboeck, Guido J. and Teuvo K. Kohonen [1998]. *Visual Explorations in Finance*.
- Delgado, Miguel, Maria J. Martín-Bautista, Daniel Sánchez, and María Amparo Vila Miranda [2002]. *Mining Text Data: Special Features and Patterns*. pages 140–153.
- Dempster, A. P., N. M. Laird, and D. B. Rubin [1977]. *Maximum Likelihood from Incomplete Data Via Em Algorithm*. *Journal of the Royal Statistical Society Series BMethodological*, 39(1), pages 1–38. ISSN 00359246.
- Demšar, J., B. Zupan, G. Leban, and T. Curk [2004]. *Orange: From Experimental Machine Learning to Interactive Data Mining*. doi:10.1007/b100704.
- Dietterich, Tom, Michael Kearns, and Yishay Mansour [1996]. *Applying the Weak Learning Framework to Understand and Improve C4.5*. IN PROCEEDINGS OF THE THIRTEENTH INTERNATIONAL CONFERENCE ON MACHINE LEARNING, pages 96 – 104.

- Dittenbach, Michael, Andreas Rauber, and Dieter Merkl [2002]. *Uncovering hierarchical structure in data using the growing hierarchical self-organizing map*. *Neurocomputing*, 48(1-4), pages 199–216. ISSN 09252312. doi:10.1016/S0925-2312(01)00655-5.
- Divina, Federico and Elena Marchiori [2002]. *Evolutionary Concept Learning*. IN *GECCO 2002: PROCEEDINGS OF THE GENETIC AND EVOLUTIONARY COMPUTATION CONFERENCE*, pages 9 – 13.
- Dong, Guozhu and Jinyan Li [1998]. *Interestingness of Discovered Association Rules in terms of Neighborhood-Based Unexpectedness*.
- Dorigo, M and L M Gambardella [1997]. *Ant colony system: a cooperative learning approach to the traveling salesman problem*. *IEEE Transactions on Evolutionary Computation*, 1(1), pages 53–66. ISSN 1089778X. doi:10.1109/4235.585892.
- Dorigo, M, V Maniezzo, and A Colorni [1996]. *Ant system: optimization by a colony of cooperating agents*. *IEEE transactions on systems man and cybernetics Part B Cybernetics a publication of the IEEE Systems Man and Cybernetics Society*, 26(1), pages 29–41. ISSN 10834419. doi:10.1109/3477.484436.
- Duda, Richard and Peter Hart [1973a]. *Pattern Classification and Scene Analysis*. 1 Edition. John Wiley & Sons Inc. ISBN 0471223611.
- Duda, Richard and Peter Hart [1973b]. *Pattern Classification and Scene Analysis*. 1 Edition. John Wiley & Sons Inc. ISBN 0471223611.
- Dutta, S. and S. Shenkar [1993]. *Bond-rating: A nonconservative application of neural networks*.
- Eiben, A E and J E Smith [2003]. *Introduction to Evolutionary Computing, Natural Computing Series*, volume 12. Springer. ISBN 3540401849, 299 pages. doi: 10.1162/evco.2004.12.2.269.
- Esposito, F., D. Malerba, G. Semeraro, and J. Kay [1997]. *A comparative analysis of methods for pruning decision trees*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5), pages 476–493. ISSN 01628828. doi:10.1109/34.589207.

- Everitt, Brian, Sabine Landau, and Morven Leese [2009]. *Cluster Analysis*. 4th Edition. Wiley. ISBN 0340761199.
- Fayyad, Usama, Gregory Piatetsky-Shapiro, and Padhraic Smyth [1996]. *The KDD process for extracting useful knowledge from volumes of data*. *Communications of the ACM*, 39(11), pages 27–34. ISSN 00010782. doi:10.1145/240455.240464.
- Fayyad, Usama M. and Keki B. Irani [1992]. *The attribute selection problem in decision tree generation*. pages 104–110.
- Ferri, César, Peter A. Flach, and José Hernández-Orallo [2002]. *Learning Decision Trees Using the Area Under the ROC Curve*. pages 139–146.
- Fortier, J. J. and H. Solomon [1996]. *Clustering procedures*. In *In proceedings of the Multivariate Analysis*, pages 493–506.
- Fraley, C [1998]. *How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis*. *The Computer Journal*, 41(8), pages 578–588. ISSN 00104620. doi:10.1093/comjnl/41.8.578.
- Frank, A. and A. Asuncion [2010]. *{UCI} Machine Learning Repository*. <http://archive.ics.uci.edu/ml>.
- Freitas, Alex A. [2002]. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*.
- Freitas, Alex A. [2003]. *A survey of evolutionary algorithms for data mining and knowledge discovery*. In Ghosh, A and S Tsutsui (Editors), *Advances in Evolutionary Computation, Natural Computing Series*, volume 136, chapter 33, pages 819–845. Springer-Verlag.
- Freitas, Alex A. [2006]. *Are We Really Discovering “Interesting ” Knowledge From Data?*
- Freitas, Alex A. and Simon Hugh Lavington [1998]. *Mining Very Large Databases With Parallel Processing*, volume 3. Springer. ISBN 0792380487, 208 pages.
- Friedman, Jerome [1991]. *Multivariate Adaptive Regression Splines*. *The Annals of Statistics*, 19(1), pages 1 – 67. ISSN 00905364. doi:10.2307/2241837.

- Friedman, Nir, Dan Geiger, and Moises Goldszmidt [1997]. *Bayesian Network Classifiers*. *Machine Learning*, 29(2-3), pages 131–163. ISSN 0885-6125. doi:10.1023/A:1007465528199.
- Friedman, Nir, Kevin Murphy, and Stuart Russell [1998]. *Learning the structure of dynamic probabilistic networks*. pages 139 – 147.
- Gamberger, Dragan, Nada Lavrac, and Viktor Jovanoski [1999]. *High Confidence Association Rules for Medical Diagnosis*.
- Garofalakis, Minos and Rajeev Rastogi [2000]. *Scalable data mining with model constraints*. *ACM SIGKDD Explorations Newsletter*, 2(2), pages 39–48. ISSN 19310145. doi:10.1145/380995.381012.
- Garofalakis, Minos N., Rajeev Rastogi, and Kyuseok Shim [1999]. *SPIRIT: Sequential Pattern Mining with Regular Expression Constraints*. pages 223–234.
- Gehrke, Johannes, Venkatesh Ganti, Raghu Ramakrishnan, and Wei-Yin Loh [1999]. *BOAT—optimistic decision tree construction*. *ACM SIGMOD Record*, 28(2), pages 169–180. ISSN 01635808. doi:10.1145/304181.304197.
- Geman, Stuart and Donald Geman [1984]. *Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6), pages 721–741. ISSN 0162-8828. doi:10.1109/TPAMI.1984.4767596.
- Ghahramani, Zoubin [1998]. *Learning dynamic Bayesian networks*. *ADAPTIVE PROCESSING OF SEQUENCES AND DATA STRUCTURES*, 1387, pages 168 – 197.
- Gluck, M A and J E Corter [1985]. *Information Uncertainty and the Utility of Categories*. In *Proceedings of the Seventh Annual Conference of Cognitive Science Society*, pages 283–287.
- Goldberg, David E. and Jon Richardson [1987]. *Genetic algorithms with sharing for multimodal function optimization*. pages 41–49.

- Guha, Sudipto, Rajeev Rastogi, and Kyuseok Shim [1998]. *CURE: An efficient clustering algorithm for large databases*. *Information Systems Journal*, 26(1), pages 73–84. ISSN 03064379. doi:10.1145/276304.276312.
- Hall, L.O., I.B. Ozyurt, and J.C. Bezdek [1999]. *Clustering with a genetically optimized approach*. *IEEE Transactions on Evolutionary Computation*, 3(2), pages 103–112. ISSN 1089778X. doi:10.1109/4235.771164.
- Han, Jiawei [2005]. *Data Mining: Concepts and Techniques*.
- Hand, David J. and Keming Yu [2001]. *Idiot’s Bayes? Not So Stupid After All?* *International Statistical Review*, 69(3), pages 385–398. ISSN 0306-7734. doi:10.1111/j.1751-5823.2001.tb00465.x.
- Handl, Julia, Julia H, and Joshua Knowles [2004]. *Evolutionary Multiobjective Clustering*. *IN PROCEEDINGS OF THE EIGHTH INTERNATIONAL CONFERENCE ON PARALLEL PROBLEM SOLVING FROM NATURE*, pages 1081 – 1091.
- Hanneman, Robert A and Mark Riddle [2005]. *Introduction to Social Network Methods*, volume 46. University of California, 5128–30 pages. doi:10.1016/j.cell.2011.03.009.
- Hastie, Trevor and Robert Tibshirani [1990]. *Generalized Additive Models*.
- Hastie, Trevor, Robert Tibshirani, and Jerome Friedman [2008]. *The Elements of Statistical Learning*. 763 pages.
- Haughton, Dominique, Joel Deichmann, Abdolreza Eshghi, Selin Sayek, Nicholas Teebagy, and Heikki Topi [2003]. *A Review of Software Packages for Data Mining*. *The American Statistician*, 57(4), pages 290–309. ISSN 00031305. doi:10.1198/0003130032486.
- He, Hongxing, Jincheng Wang, Warwick Graco, and Simon Hawkins [1997]. *Application of neural networks to detection of medical fraud*. *Expert Systems with Applications*, 13(4), pages 329–336. ISSN 09574174. doi:10.1016/S0957-4174(97)00045-6.
- Hekanaho, Jukka [1996]. *Testing Different Sharing Methods in Concept Learning*.

- Hilderman, Robert J. and Howard J. Hamilton [2001]. *Knowledge Discovery and Measures of Interest*.
- Holmes, G, A Donkin, and I H Witten [1994]. *WEKA: a machine learning workbench*. doi:10.1109/ANZIIS.1994.396988.
- Holzinger, A, K H Struggl, and M Debevc [2010]. *Applying Model-View-Controller (MVC) in design and development of information systems: An example of smart assistive script breakdown in an e-Business application*.
- Holzinger, Andreas [2005]. *Usability engineering methods for software developers*. *Communications of the ACM*, 48(1), pages 71–74. ISSN 00010782. doi:10.1145/1039539.1039541.
- Holzinger, Andreas [2010]. *Process Guide for Students for Interdisciplinary Work in Computer Science/Informatics. Second Edition*. BoD, Norderstedt, 128 pages. <http://hci4all.at/projects4students.html>http://books.google.at/books?id=37-V3UmRe_IC&lpg=PP1&dq=ProcessGuideforStudentsforInterdisciplinaryWorkinComputerScience/Informatics&pg=PP1#v=onepage&q&f=false.
- Holzinger, Andreas [2012a]. *Biomedical Informatics: Computational Sciences meets Life Sciences*. http://genome.tugraz.at/medical_informatics.shtml.
- Holzinger, Andreas [2012b]. *On Knowledge Discovery and Interactive Intelligent Visualization of Biomedical Data - Challenges in Human-Computer Interaction & Biomedical Informatics*. In *DATA*.
- Holzinger, Andreas, Martin Brugger, and Wolfgang Slany [2011]. *APPLYING ASPECT ORIENTED PROGRAMMING IN USABILITY ENGINEERING PROCESSES On the example of Tracking Usage Information for Remote Usability Testing*. In Marca, David A, B Shishkov, and M V Sinderen (Editors), *Proceedings of the 8th International Conference on electronic Business and Telecommunications*, pages 53–56. SciTePress INSTICC. ISBN 9789898425706.
- Hopfield, J J [1982]. *Neural networks and physical systems with emergent collective computational abilities*. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8), pages 2554–2558.

- Hu, X., J. Zhang, and Y. Li [2008]. *Orthogonal methods based ant colony search for solving continuous optimization problems*. doi:10.1007/s11390-008-9111-5<<http://dx.doi.org/10.1007/s11390-008-9111-5>>).
- Imielinski, Tomasz and Heikki Mannila [1996]. *A database perspective on knowledge discovery*. *Communications of the ACM*, 39(11), pages 58–64. ISSN 00010782. doi:10.1145/240455.240472.
- Janikow, C Z [1998]. *Fuzzy decision trees: issues and methods*. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 28(1), pages 1–14. ISSN 1083-4419. doi: 10.1109/3477.658573.
- Jensen, J L and W S Kendall [1993]. *Networks and Chaos - Statistical and Probabilistic Aspects*. Monographs on Statistics and Applied Probability, Taylor & Francis. ISBN 9780412465307.
- Jeudy, Baptiste, Jean-Francois Boulicaut, and Btiment Blaise Pascal [2002]. *Optimization of Association Rule Mining Queries*. *INTELLIGENT DATA ANALYSIS*, 6, pages 341 – 357.
- Jiao, Licheng Jiao Licheng, Jing Liu Jing Liu, and Weicai Zhong Weicai Zhong [2006]. *An organizational coevolutionary algorithm for classification*. doi:10.1109/TEVC.2005.856068.
- Jing, Yushi and Shumeet Baluja [2008]. *Pagerank for product image search*. In *Proceeding of the 17th international conference on World Wide Web - WWW '08*, page 307. ACM Press, New York, New York, USA. ISBN 9781605580852. doi: 10.1145/1367497.1367540.
- Joachims, Thorsten [2002]. *Learning to Classify Text Using Support Vector Machines. Methods, Theory and Algorithms*. 668.
- Jong, Kenneth A De [2006]. *Evolutionary computation: a unified approach*, volume 8. MIT Press. ISBN 0262041944, 2811–2824 pages. doi:10.1007/s10710-007-9035-9.
- Kaufman, L and P Rousseeuw [1987]. *Clustering by Means of Medoids*. Reports of the Faculty of Mathematics and Informatics. Delft University of Technology, Fac., Univ.

- Kearns, Michael and Yishay Mansour [1996]. *On the boosting ability of top-down decision tree learning algorithms*. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*, pages 459–468. ACM Press, New York, New York, USA. ISBN 0897917855. doi:10.1145/237814.237994.
- Kennedy, J. and R. Eberhart [1995]. *Particle swarm optimization*. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948. IEEE. ISBN 0-7803-2768-3. doi:10.1109/ICNN.1995.488968.
- Kianmehr, Keivan, Mehmet Kaya, Abdallah M ElSheikh, Jamal Jida, and Reda Alhajj [2011]. *Fuzzy association rule mining framework and its application to effective fuzzy associative classification*. *Wiley Interdisciplinary Reviews Data Mining and Knowledge Discovery*, 1(6), pages 477–495. ISSN 19424787. doi:10.1002/widm.40.
- Kim, YeongSeog, W. Nick Street, and Filippo Menczer [2000]. *Feature selection in unsupervised learning via evolutionary search*. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*, pages 365–369. ACM Press, New York, New York, USA. ISBN 1581132336. doi:10.1145/347090.347169.
- King, B [1967]. *Step-wise clustering procedures*. *Journal of the American Statistical Association*, 69(317), pages 86–101. ISSN 01621459. doi:10.2307/2282912.
- Kleinberg, Jon M [1999]. *Authoritative sources in a hyperlinked environment*. *Journal of the ACM*, 46(5), pages 604–632. ISSN 00045411. doi:10.1145/324133.324140.
- Kohonen, T [1990]. *The self-organizing map*. *Proceedings of the IEEE*, 78(9), pages 1464–1480. ISSN 00189219. doi:10.1109/5.58325.
- Korkmaz, Emin Erkan, Jun Du, Reda Alhajj, and Ken Barker [2006]. *Combining advantages of new chromosome representation scheme and multi-objective genetic algorithms for better clustering*. *Intelligent Data Analysis*, 10(2), pages 163–182. ISSN 1088-467X.
- Kotsiantis, S B [2007]. *Supervised Machine Learning : A Review of Classification Techniques*. *Informatika*, 31(3), pages 249–268. ISSN 09226389.

- Kramer, Stefan, Luc De Raedt, and Christoph Helma [2001]. *Molecular feature mining in HIV data*. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '01*, pages 136–143. ACM Press, New York, New York, USA. ISBN 158113391X. doi:10.1145/502512.502533.
- Kressel, Ulrich H.-G. [1999]. *Pairwise classification and support vector machines*. pages 255–268.
- Langley, Pat, Iba, , and Kevin Thompson [1992]. *An analysis of Bayesian classifiers*. pages 223–228.
- Larranaga, P., C.M.H. Kuijpers, R.H. Murga, and Y. Yurramendi [1996]. *Learning Bayesian network structures by searching for the best ordering with genetic algorithms*. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 26(4), pages 487–493. ISSN 10834427. doi:10.1109/3468.508827.
- Lauritzen, S. L. and D. J. Spiegelhalter [1990]. *Local computations with probabilities on graphical structures and their application to expert systems*. pages 415–448.
- Lauritzen, Steffen [1996]. *Graphical Models (Oxford Statistical Science Series)*. Oxford University Press, USA. ISBN 0198522193.
- Lewis, O. M., J. A. Ware, and D. Jenkins [1997]. *A novel neural network technique for the valuation of residential property*. *Neural Computing & Applications*, 5(4), pages 224–229. ISSN 0941-0643. doi:10.1007/BF01424227.
- Lim, Loh, and Shih [2000]. *A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms*. *Machine Learning*, 40(3), pages 203–228. ISSN 08856125. doi:10.1023/A:1007608224229.
- Liu, Bing, Wynne Hsu, and Shu Chen [1997]. *Using General Impressions to Analyze Discovered Classification Rules*.
- Liu, Huan and Hiroshi Motoda [2001]. *Instance Selection and Construction for Data Mining*.
- Liu, J Juan and J Tin-Yau Kwok [2000]. *An extended genetic rule induction algorithm*. doi:10.1109/CEC.2000.870332.

- Lobo, Fernando G. and David E. Goldberg [2004]. *The parameter-less genetic algorithm in practice*. *Information Sciences*, 167(1-4), pages 217–232. ISSN 00200255. doi:10.1016/j.ins.2003.03.029.
- Lockwood, Lucy [2003]. *Usability by Inspection: The Collaborative Approach*.
- Loh, WY and YS Shih [1997]. *Split Selection Methods for Classification Trees*. *Statistica Sinica*.
- Lu, Qing and Lise Getoor [2003]. *Link-based Text Classification*.
- Maimon, Oded and Lior Rokach [2010]. *Data Mining and Knowledge Discovery Handbook*. Springer, 1277 pages.
- Mannila, Heikki and Hannu Toivonen [1996]. *Multiple uses of frequent sets and condensed representations (Extended Abstract)*. *IN PROC. KDD INT. CONF. KNOWLEDGE DISCOVERY IN DATABASES*, pages 189 – 194.
- Mannila, Heikki and Hannu Toivonen [1997]. *Levelwise search and borders of theories in knowledge discovery*. *Data Mining and Knowledge Discovery*, 1(3), pages 241–258. ISSN 13845810.
- Mannila, Heikki, Hannu Toivonen, and A Inkeri Verkamo [1997]. *Discovery of Frequent Episodes in Event Sequences*. *Data Mining and Knowledge Discovery*, 1(3), pages 259–289. ISSN 13845810. doi:10.1023/A:1009748302351.
- Martin, David R, Charless C Fowlkes, and Jitendra Malik [2004]. *Learning to detect natural image boundaries using local brightness, color, and texture cues*. *IEEE transactions on pattern analysis and machine intelligence*, 26(5), pages 530–49. ISSN 0162-8828. doi:10.1109/TPAMI.2004.1273918.
- Merrill, Jacqueline and George Hripcsak [2008]. *Using social network analysis within a department of biomedical informatics to induce a discussion of academic communities of practice*. *Journal of the American Medical Informatics Association : JAMIA*, 15(6), pages 780–2. ISSN 1067-5027. doi:10.1197/jamia.M2717.
- Millonas, Mark M [1992]. *Swarms, phase transitions, and collective intelligence*. *Intelligence*, pages 1–32.

- Mingers, John [1989]. *An Empirical Comparison of Pruning Methods for Decision Tree Induction*. *Machine Learning*, 4(2), pages 227–243. ISSN 0885-6125. doi: 10.1023/A:1022604100933.
- Murtagh, F [1983]. *A Survey of Recent Advances in Hierarchical Clustering Algorithms*. *The Computer Journal*, 26(4), pages 354–359. ISSN 09201211. doi: 10.1093/comjnl/26.4.354.
- Nadkarni, Sucheta and Prakash P Shenoy [2001]. *A Bayesian network approach to making inferences in causal maps*. *European Journal of Operational Research*, 128(3), pages 479–498. ISSN 03772217. doi:10.1016/S0377-2217(99)00368-9.
- Negulescu, Sorin C., Constantin Oprean, Claudiu V. Kifor, and Ilie Carabulea [2008]. *Elitist ant system for route allocation problem*. pages 62–67.
- Neville, Jennifer, Micah Adler, and David Jensen [2003]. *Clustering Relational Data Using Attribute and Link Information*. *Proceedings of the text mining and link analysis workshop 18th international joint conference on artificial intelligence*, pages 9–15.
- Ng, Raymond T, Laks V S Lakshmanan, Jiawei Han, and Alex Pang [1998]. *Exploratory mining and pruning optimizations of constrained associations rules*. *Proceedings of the 1998 ACM SIGMOD international conference on Management of data SIGMOD 98*, 27(2), pages 13–24. ISSN 01635808. doi:10.1145/276304.276307.
- Olaru, Cristina and Louis Wehenkel [2003]. *A complete fuzzy decision tree technique*. *Fuzzy Sets and Systems*, 138(2), pages 221–254. ISSN 01650114. doi:10.1016/S0165-0114(03)00089-7.
- Padmanabhan, Balaji and Alexander Tuzhilin [1998]. *A Belief-Driven Method for Discovering Unexpected Patterns*. pages 94 – 100.
- Pagallo, Giulia and David Haussler [1990]. *Boolean Feature Discovery in Empirical Learning*. *Machine Learning*, 5(1), pages 71–99. ISSN 0885-6125. doi:10.1023/A:1022611825350.

- Page, Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd [1998]. *The PageRank Citation Ranking: Bringing Order to the Web*. *World Wide Web Internet And Web Information Systems*, 54(2), pages 1–17.
- Paik, J K and A K Katsaggelos [1992]. *Image restoration using a modified Hopfield network*. doi:10.1109/CERMA.2007.4367691.
- Pasquier, Nicolas, Yves Bastide, Rafik Taouil, and Lotfi Lakhal [1999]. *Efficient mining of association rules using closed itemset lattices*. *Information Systems*, 24(1), pages 25–46. ISSN 03064379. doi:10.1016/S0306-4379(99)00003-4.
- Pearl, Judea [1988]. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. *Morgan Kauffmann San Mateo (1988)*, 88(8), page 552. doi:10.2307/2026705.
- Piatetsky-Shapiro, Gregory [1991]. *Discovery, Analysis, and Presentation of Strong Rules*. In Piatetsky-Shapiro, Gregory and William J Frawley (Editors), *Knowledge Discovery in Databases*, volume 229, pages 229–248. AAAI/MIT Press. ISBN 0262620804.
- Quinlan, J R [1986]. *Induction of decision trees*. *Machine Learning*, 1(1), pages 81–106. ISSN 1062936X. doi:10.1080/10629360600933723.
- Quinlan, J R [1987]. *Simplifying decision trees*. *International Journal of ManMachine Studies*, 27(3), pages 221–234. ISSN 00207373. doi:10.1016/S0020-7373(87)80053-6.
- Quinlan, J R [1993]. *C4.5: Programs for Machine Learning, Morgan Kaufmann series in {M}achine {L}earning*, volume 1. Morgan Kaufmann. ISBN 1558602380, 302 pages.
- Ramoni, Marco and Paola Sebastiani [2003]. *Bayesian Methods*.
- Rand, M [1971]. *Objective Criteria for the Evaluation of Methods Clustering*. *Journal of the American Statistical Association*, 66(336), pages 846–850. ISSN 01621459. doi:10.1007/BF01908075.

- Reinartz, Thomas [2002]. *A Unifying View on Instance Selection*. *Data Mining and Knowledge Discovery*, 6(2), pages 191–210. ISSN 1384-5810. doi:10.1023/A:1014047731786.
- Romão, Wesley, Alex A Freitas, and Itana M De S. Gimenes [2004]. *Discovering interesting knowledge from a science & technology database with a genetic algorithm*. *IN APPLIED SOFT COMPUTING 4*, pages 121 – 137.
- Ruspini, E H, J Thomere, and M Wolverson [2004]. *Database editing metrics for pattern matching*. doi:10.1109/CIHSPS.2004.1360204.
- Sahami, Mehran [1996]. *Learning Limited Dependence Bayesian Classifiers*. *IN KDD-96: PROCEEDINGS OF THE SECOND INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING*, pages 335 – 338.
- Scholkopf, Bernhard and Alexander J. Smola [2001]. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*.
- Senator, T. E., H. G. Goldberg, J Wooton, A Cottini, A Umar, C Klinger, W Llamas, M Marrone, and R Wong [1995]. *The fincen artificial intelligence system: Identifying potential money laundering from reports of large cash transactions*. pages 283–302.
- Seni, Giovanni and John F. Elder [2010]. *Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions*.
- Shachter, R. D. [1986]. *Evaluating Influence Diagrams*. *Operations Research*, 34(6), pages 871–882. ISSN 0030-364X. doi:10.1287/opre.34.6.871.
- Singh, Moninder and Marco Valtorta [1995]. *Construction of Bayesian Network Structures from Data: a Brief Survey and an Efficient Algorithm*. *International Journal Of Approximate Reasoning*, 12, pages 259 – 265.
- Smith, Kate A. and Alan Ng [2003]. *Web page clustering using a self-organizing map of user navigation patterns*. *Decision Support Systems*, 35(2), pages 245–256. ISSN 01679236. doi:10.1016/S0167-9236(02)00109-4.
- Smyth, P and R M Goodman [1992]. *An information theoretic approach to rule induction from databases*. doi:10.1109/69.149926.

- Sokal, R R and P H A Sneath [1963]. *Principles of Numerical Taxonomy, A Series of books in biology*, volume 57. Freeman. ISBN 9780716706212, 359 pages.
- Srikant, Ramakrishnan, Quoc Vu, and Rakesh Agrawal [1997]. *Mining Association Rules with Item Constraints*. pages 67 – 73.
- Steinwart, Ingo [2003]. *Sparseness of support vector machines*. *The Journal of Machine Learning Research*, 4, pages 1071–1105. ISSN 1532-4435.
- Strehl, Alexander and Joydeep Ghosh [2000]. *Clustering Guidance And Quality Evaluation Using Relationship-Based Visualization*.
- Stützle, Thomas and Holger H Hoos [2000]. *MAX-MIN Ant system*. *Future Gener Comput Syst*, 16(9), pages 889–914. ISSN 0167739X. doi:10.1016/S0167-739X(00)00043-1.
- Taskar, Ben [2001]. *Probabilistic classification and clustering in relational data*. IN *PROCEEDINGS OF THE SEVENTEENTH INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 870 – 878.
- Tatem, A.J., H.G. Lewis, P.M. Atkinson, and M.S. Nixon [2002]. *Super-resolution land cover pattern prediction using a Hopfield neural network*. doi:10.1016/S0034-4257(01)00229-2<[http://dx.doi.org/10.1016/S0034-4257\(01\)00229-2](http://dx.doi.org/10.1016/S0034-4257(01)00229-2)>).
- Tibshirani, Robert, Guenther Walther, and Trevor Hastie [2001]. *Estimating the number of clusters in a data set via the gap statistic*. *Journal of the Royal Statistical Society - Series B: Statistical Methodology*, 63(2), pages 411–423. ISSN 13697412. doi:10.1111/1467-9868.00293.
- Tsakonas, Athanasios, Georgios Dounias, Jan Jantzen, Hubertus Axer, Beth Bjerregaard, and Diedrich Graf von Keyserlingk [2004]. *Evolving rule-based systems in two medical domains using genetic programming*. *Artificial intelligence in medicine*, 32(3), pages 195–216. ISSN 0933-3657. doi:10.1016/j.artmed.2004.02.007.
- Tsang, Ivor W., Andras Kocsor, and James T. Kwok [2007]. *Simpler core vector machines with enclosing balls*. In *Proceedings of the 24th international conference*

- on *Machine learning - ICML '07*, pages 911–918. ACM Press, New York, New York, USA. ISBN 9781595937933. doi:10.1145/1273496.1273611.
- Tsang, Ivor W., James T. Kwok, and Pak-Ming Cheung [2005a]. *Core Vector Machines: Fast SVM Training on Very Large Data Sets*. *The Journal of Machine Learning Research*, 6, pages 363–392. ISSN 1532-4435.
- Tsang, Ivor W., James T. Kwok, and Kimo T. Lai [2005b]. *Core Vector Regression for very large regression problems*. In *Proceedings of the 22nd international conference on Machine learning - ICML '05*, pages 912–919. ACM Press, New York, New York, USA. ISBN 1595931805. doi:10.1145/1102351.1102466.
- Tsang, Ivor W., Ivor W Tsang, and James T Kwok [2006]. *Large-scale sparsified manifold regularization*. *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS (NIPS) 19*.
- Urquhart, Roderick [1982]. *Graph theoretical clustering based on limited neighbourhood sets*. *Pattern Recognition*, 15(3), pages 173–187. ISSN 00313203. doi:10.1016/0031-3203(82)90069-3.
- Vapnik, Vladimir N. [1995]. *The nature of statistical learning theory*.
- Wallace, C. S. and J. D. Patrick [1993]. *Coding Decision Trees*. *Machine Learning*, 11(1), pages 7–22. ISSN 0885-6125. doi:10.1023/A:1022646101185.
- Wallace, Christopher S., Christopher S Wallace, and David L Dowe [1994]. *Intrinsic Classification by MML—the Snob Program*. *PROC. SEVENTH AUSTRALIAN JOINT CONF. ARTIFICIAL INTELLIGENCE*, 17, pages 37 – 44.
- Wang, Jianyong, Jiawei Han, and Jian Pei [2003]. *CLOSET+*. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '03*, page 236. ACM Press, New York, New York, USA. ISBN 1581137370. doi:10.1145/956750.956779.
- Ward, J H [1963]. *Hierarchical grouping to optimize an objective function*. *Journal of the American Statistical Association*, 58(301), pages 236–244. ISSN 01621459. doi:10.2307/2282967.

- Wass, John A [2007]. *CART: A Powerful Decision Tree Tool for Predictive Modeling*. *Scientific Computing*, 24(7), pages 47–49. ISSN 19305753.
- Wasserman, Stanley, Katherine Faust, and Dawn Iacobucci [1994]. *Social Network Analysis : Methods and Applications (Structural Analysis in the Social Sciences)*.
- Webb, Geoffrey I. [2000]. *Efficient search for association rules*. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*, pages 99–107. ACM Press, New York, New York, USA. ISBN 1581132336. doi:10.1145/347090.347112.
- Wei, Chu [2003]. *Bayesian Approach To Support Vector Machines*.
- Whittaker, Joe [1990]. *Graphical Models in Applied Multivariate Statistics*.
- Witten, Ian H. and Eibe Frank [2005]. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*.
- Wood, Simon N [2004]. *Stable and Efficient Multiple Smoothing Parameter Estimation for Generalized Additive Models*. *Journal of the American Statistical Association*, 99(467), pages 673–686. ISSN 0162-1459. doi:10.1198/016214504000000980.
- Wu, Xindong and Vipin Kumar [2009]. *The Top Ten Algorithms in Data Mining*.
- Wyns, B, S Sette, L Boullart, D Baeten, I E A Hoffman, and F De Keyser [2004]. *Prediction of diagnosis in patients with early arthritis using a combined Kohonen mapping and instance-based evaluation criterion*. *Artificial intelligence in medicine*, 31(1), pages 45–55. ISSN 0933-3657. doi:10.1016/j.artmed.2004.01.002.
- Yin, Peng-Yeng [1998]. *A new method for polygonal approximation using genetic algorithms*. *Pattern Recognition Letters*, 19(11), pages 1017–1026. ISSN 01678655. doi:10.1016/S0167-8655(98)00082-8.
- Yu, Lei and Huan Liu [2004]. *Efficient Feature Selection via Analysis of Relevance and Redundancy*. *The Journal of Machine Learning Research*, 5, pages 1205–1224. ISSN 1532-4435.

- Zahn, C T [1971]. *Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters*. doi:10.1109/T-C.1971.223083.
- Zaki, Mohammed J. [2000]. *Generating non-redundant association rules*. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*, pages 34–43. ACM Press, New York, New York, USA. ISBN 1581132336. doi:10.1145/347090.347101.
- Zhang, Qingyu [2010]. *Review of data, text and web mining software*. *Kybernetes*, 39(4).
- Zhang, Shichao, Chengqi Zhang, and Qiang Yang [2003]. *Data preparation for data mining*. *APPLIED ARTIFICIAL INTELLIGENCE*, 17, pages 375 – 381.
- Zhou, Chi Zhou Chi, Weimin Xiao Weimin Xiao, T M Tirpak, and P C Nelson [2003]. *Evolving accurate and compact classification rules with gene expression programming*. doi:10.1109/TEVC.2003.819261.
- Zwitter, M and M Soklic [1988]. *Breast Cancer Characteristics and Recurrence Data*.