Aalto University
School of Science and Technology
Faculty of Electronics, Communications and Automation
Degree Programme of Computer Science and Engineering

Sami Hänninen

Applying data mining techniques to ERP system anomaly and error detection

Master's Thesis

Espoo, April 8, 2010

Supervisor: Professor Antti Ylä-Jääski

Instructors: Timo Kiravuo, M. Sc.
Olli Veikkola, M. Sc.

Aalto University
School of Science
and Technology

ABSTRACT OF THE MASTER'S THESIS

Aalto University School of Science and Technology Faculty of Electronics, Communications and Automation Degree Programme of Computer Science and Engineering		
Author: Sami Hänninen		
Title: Applying data mining techniques to ERP system anomaly and error detection		
Number of Pages: 76	Date: 8.4.2010	Language: English
Professorship: Computer Science and Engineering		Code: T-110
Supervisor: Professor Antti Ylä-Jääski		
Instructors: Timo Kiravuo, M.Sc., Olli Veikkola, M.Sc.		
Abstract: Data mining is a concept developed for analyzing large quantities of data. It is based on machine learning, pattern recognition and statistics and is currently used, for example, for fraud detection, marketing advice, predicting sales and inventory and correcting data. Enterprise Resource Planning (ERP) systems commonly gather data from all parts of company operations thus providing a good source of data for data mining. This thesis studies data mining suitability for real-time validation of Lean System ERP input on the Oracle Data Mining (ODM) platform to improve data quality. The results proved that data mining can be a successful tool for input validation, but a successful mining process requires often meticulous preprocessing of mined data and good knowledge of the algorithms.		
Keywords: data mining, database, ERP, KDD, input validation, Oracle, Lean System, ODM, data quality		

Aalto-yliopisto
Teknillinen korkeakoulu

DIPLOMITYÖN TIIVISTELMÄ

Aalto-yliopisto Teknillinen korkeakoulu Elektroniikan, tietoliikenteen ja automaation tiedekunta Tietotekniikan tutkinto-ohjelma/koulutusohjelma		
Tekijä: Sami Hänninen		
Työn nimi: Tiedonlouhinnan soveltaminen ERP-järjestelmän poikkeamien ja virheiden havaitsemiseen		
Sivumäärä: 76	Pvm: 8.4.2010	Julkaisukieli: englanti
Professuuri: Tietotekniikan laitos		Professuurikoodi: T-110
Valvoja: Professori Antti Ylä-Jääski		
Ohjaajat: Timo Kiravuo, DI, Olli Veikkola, DI		

Tiivistelmä: Tiedonlouhinta kehitettiin oleellisen tiedon löytämiseen suurista tietomassoista ja pohjautuu koneoppimiseen, hahmontunnistukseen ja tilastotieteeseen. Suosittuja käyttökohteita ovat esimerkiksi huijausten havainnointi, markkinointianalyysit, myynnin ja varaston ennustaminen sekä tietojen korjaus. Toiminnanohjausjärjestelmät (ERP) keräävät usein suuria määriä tietoja kaikista yrityksen toiminnoista, mikä tekee niistä hyvän kohteen tiedonlouhinnalle. Tämä diplomityö tutkii tiedonlouhinnan sopimista Lean System toiminnanohjausjärjestelmän syötteiden tarkistukseen tosiaikaisesti Oraclen tiedonlouhinta-alustalla tiedon laadun parantamiseksi. Tulokset osoittavat, että tiedonlouhinta voi olla menestyksekäs työkalu syötteen tarkistukseen, mutta onnistunut louhintaprosessi vaatii usein louhittavan tiedon pikkutarkkaa esikäsittelyä ja algoritmien hyvää tuntemusta.

Avainsanat: tiedonlouhinta, tietokanta, ERP, KDD, syötteen tarkistus, Oracle, Lean System, ODM, tiedon laatu

ACKNOWLEDGEMENTS

I would like to thank my supervisor Professor Antti Ylä-Jääski and instructors M.Sc. Timo Kiravuo and M.Sc. Olli Veikkola for guidance and feedback during this work. I would also like to thank Professor Kari Halonen for giving me the opportunity to do this thesis about a minor study subject and for his excellent lectures during my longish studies. I am grateful to Taina Hyppölä, Sanna Suoranta and especially Pirjo Putila from Aalto University for giving guidance and help on finalizing my studies and to Risto Raunio and Harri Sandqvist from Tieto for support. I also wish to thank William Martin from Aalto University for proofreading the final manuscript of this thesis.

Invaluable knowledge from a project manager's viewpoint was given by Project Manager Bodil Granskog from Tieto. Other persons contributing were Mr. Rafal Lukawiecki from Project Botticelli Ltd. by giving me the initial enthusiasm for this subject with his capturing presentations and Marko Kosunen from Aalto University, Department of Micro and Nanosciences for showing great patience and understanding when faced with special assignment delays. I am also in gratitude to the company that gave me permission to use their production data for this study and to Aalto University Library for excellent service.

Finally, I also want to thank my wife Pia for encouraging me to write and suggesting corrections, and to my children, Lenni and Ernesti, for just being there as a counterbalance and preventing me from writing.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

ABBREVIATIONS

ADP	Automatic Data Preparation – automatic data set preprocessing functionality in Oracle Data Mining software.
AP	Apriori – data mining algorithm for association analysis.
API	Application Programming Interface – interface the software publishes for other programs to interact with it.
BI	Business Intelligence – bases itself on KDD and tries to improve business decision making by using fact-based support systems.
BIRCH	Balanced Iterative Reducing and Clustering using Hierarchies – hierarchical clustering algorithm.
CRISP-DM	CRoss-Industry Standard Process for Data Mining – data mining methodology created by a consortium founded for developing a generalized data mining process.
CRM	Customer Relationship Management – concepts and programs used by companies to manage customer relationships.
DBSCAN	Density-Based Spatial Clustering of Applications with Noise – local density minimum based clustering algorithm.
DM	Data Mining – process of extracting useful information from such large masses of data that it is not possible with statistical analysis.
DW	Data Warehouse – a repository for electronically stored data gathered from multiple sources.
ERP	Enterprise Resource Planning - software system or framework for managing all operations of a company. Examples of software suites are Lean System, SAP, Baan, etc.
ETL	Extract-Transform-Load operation – operation where data is extracted from a source, transformed and loaded into target database.
GLM	Generalized Linear Model – data mining algorithm suitable for regression and classification, based on linear regression.
GUI	Graphical User Interface – interface based on icons, pictures, menus etc. for user to interact with a system, contains often display and several input mechanisms like mouse and keyboard.
IDS	Intrusion Detection System – network intrusion detection software which tries to automate the detection and possible response.

IEC	International Electrotechnical Commission – standards organization for electrical, electronic and related technologies.
IPMAP	Information Product Mapping – systematic representation of information product as a map.
ISO	International Organization for Standardization – standardizes and publishes industrial and commercial standards.
JIT	Just In Time - operations management philosophy based on low in-process inventories, cross training of workers, flexibility of equipment.
KDD	Knowledge Discovery in Databases – process of converting raw data into useful information with DM, includes data pre- and post-processing.
KM	K-Means – data mining algorithm for clustering, based on distance measurements between cases.
MDL	Minimum Description Length – data mining algorithm suitable for feature selection.
MIT	Massachusetts Institute of Technology – university in Cambridge, Massachusetts, USA.
O-cluster	Orthogonal Partitioning Clustering – scalable clustering algorithm for large high dimensional data sets, based on density concentrations.
ODM	Oracle Data Miner – Oracle's GUI for Data Mining functionality.
PCM	Pulse Code Modulation – method for encoding audio signals to digital format by sampling them.
PL/SQL	Procedural Language/Structured Query Language – Oracle Corporation's proprietary procedural extension to SQL.
RMSE	Root Mean Squared Error – square root of the average squared distance of a data point from the regression line.
ROC	Receiver Operating Characteristics – metric for comparing classification predictions to actual values.
SCM	Supply Chain Management - concept that helps companies to manage the whole supply chain from raw materials to consumption over organizational and company borders.
SQL	Structured Query Language – language for querying and updating databases.
SVM	Support Vector Machines – data mining algorithm for classification, based on statistical learning theory.

VTKK	Valtion Tietokonekeskus – central institution for state administration's data processing. Was later changed from state-owned business to private company and finally bought by Tietotehdas, later Tieto Oyj.
WEKA	Waikato Environment for Knowledge Analysis – collection of machine learning algorithms for data mining tasks.

1 INTRODUCTION

The software systems used in today's corporate world create and gather data in many forms. The data originates from multiple input points and multiple inputs, it has many sources and consists for example of orders, invoices, purchases, manufacturing data, measurements and anything else that is linked to running the business of a company. This information is stored in massive databases but is mostly used in day to day operations without largely analyzing the data and finding additional value from it. This stems from the fact that finding useful information is challenging due to the large volume of the data or the nature of the data might make basic statistical analysis impossible. (Tan et al., 2006)

Data Mining (DM) is a concept developed for analyzing these large quantities of data. Hand, Mannila and Smyth gave a good definition for DM as "the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner" (Hand et al., 2001). This signifies that DM can be used to analyze large data masses to find essential information which cannot be seen directly from the data by means of, for example, statistical analysis.

DM has its origins in machine learning, pattern recognition and statistics. It was developed in the 1980's when representatives of several research areas noticed that they needed more sophisticated algorithms in addition to classical statistical analysis to overcome problems in analyzing large quantities of data (Piatetsky-Shapiro, 1991). The motivation for DM came from several challenging problems including scalability, high dimensionality heterogeneous and complex data, data ownership as well as distribution and non-traditional analysis (Tan et al., 2006). Knowledge Discovery in Databases (KDD) is sometimes used as a synonym for DM, but, as a wider concept, it denotes a process which tries to convert raw data into useful information using DM, and also includes pre- and post processing phases in addition to the actual data mining (Han & Kamber, 2001).

Enterprise Resource Planning (ERP) software is often one of the extensive information gatherers in a company. It can be used as a managing system for various parts of business, including product or production planning, parts purchasing, inventory tracking, supply chain and order management (Uusi-Rauva et al., 1994). Some commonly known examples of ERP systems are SAP, Baan, MFG/Pro, Oracle eBusiness Suite and Microsoft Dynamics.

ERP systems always store their data in a database which makes it a viable target for data mining activities, but many times companies use diverse systems together to form complete ERP functionality and their data is scattered across multiple databases. In these cases, it is sometimes necessary to gather the scattered data into a single database called a Data Warehouse (DW), before submitting it to data mining activity.

The test case selected for this thesis is Lean System ERP by Tieto Corporation Oyj. One of the main development themes and working principles of this ERP is the freedom of choice to continue work without complete information and bringing decision making based on incomplete knowledge to the user, letting the system take care of the routine operations. Uncertainty and errors are caused by the missing information and guessed values, and these are the kind of things DM could be used to tackle. Input verification is also a critical issue to avoid unnecessary problems during the processes involved, for instance, to avoid unwanted price changes. (Veikkola, 2009; Tieto Corporation Oyj, 2009) Real-life customer data from one manufacturing area is used in this research.

1.1 PROBLEM STATEMENT

This thesis studies the application of artificial intelligence to an ERP system to notice if something out of the ordinary happens without hard-coding any rules to notice these anomalies (Lukawiecki, 2008). This falls into the anomaly detection category of DM and there are some algorithms covered later for this application. Some working examples of anomaly detection are network intrusion detection (IDS) systems and fraud detection in credit card purchases.

While an ERP system controls the daily operations of a company, an ERP's performance depends largely on the quality of the data, as can be seen from data quality research discussed later. This Master's thesis attempts to solve the problem of maintaining that data quality by input verification. The data can be erroneous or out of feasible range and it might even be transferred to the system automatically from an outside source (Granskog & Veikkola, 2009; Lukawiecki, 2008). With the data volumes present in an ERP system, it is impossible to verify all information by hand or strict rules, so an automatic method of error correction, or at least error notification, could prevent ERP performance degradation.

The research question this thesis strives to answer is: Is it possible to find an erroneous data entry in an Enterprise Resource Planning system input without hard-coding the rules using data mining algorithms to avoid data quality degradation and problems caused by it?

There are some additional problems related to this research question and they consist of possibly finding some common data input problems and static rules to avoid them, and making use of ready-made DM algorithms in an Oracle database to answer the research question. There are other DM software manufacturers even solely concentrated on the DM concept, but these are not evaluated since the functionality has to be built-in with the database already in use. The results might still apply to these systems as well, since the algorithms are not generally proprietary but well-known.

The objective is to find frequently adjustable algorithms which can identify problematic data even when the statistical nature of the data changes over time. It is not yet known if this is possible at all with Lean System ERP data sets, but if successful candidates are found, they might be implemented and included in the Lean System product in the future.

The research question is divided into a few steps. The first one is identifying some problematic areas in ERP data by interviewing project managers and customer support analysts. After this, suitable candidate problems found are being targeted for closer inspection to find an accurate problem statement for the data mining process. Blind testing some algorithms with the data can be also tried to explore the problem area, but according to Fayyad et al. (1996b) this can lead to false discoveries. When the definition of the problem statement is clear, source data set, methods and algorithms are being evaluated and tested to find possible working solutions and improvement targets to enable this functionality in the future. If acceptable DM solutions or static rules are found as an output from the whole DM process, their implementation can be researched as future work.

2 BACKGROUND AND EARLIER RESEARCH

This chapter contains descriptions of entities, concepts, methods and algorithms related to this work, together with notable earlier research.

2.1 ENTERPRISE RESOURCE PLANNING

Enterprise Resource Planning (ERP) software suites try to incorporate all operational management functionality needed in an enterprise, such as material requirement planning, the decision support system and the management information system (Shtub, 2002). More functionality is added regularly to cover new aspects of operations management, depending on the current trend for organizational development. Some of the newest development targets have been, for example, Customer Relationship Management (CRM), cross-organizational ERP and Supply Chain Management (SCM), even though they all have been around for at least several years as concepts. (Bayraktar et al., 2007)

ERP systems are based on several operations management philosophies, from which the current trends seem to be Lean Production, Agile Manufacturing and mass customization. Lean Production tries to eliminate waste through optimizing value stream for a product. This is achieved by eliminating unnecessary storage, transportation and inspection. Agile Manufacturing bases itself on succeeding under constantly changing circumstances by managing, for example, people and information and accepting change as a logical part of the process. Finally, mass customization attempts to combine benefits of mass production and high quality, low volume customized production, but apparently in Finland this philosophy is not very common. Mass customization is anyway achieved by postponing the customization phases to the finalization of a product. (Womack et al., 1991; Zhang & Sharifi, 2000; Granskog & Veikkola, 2009)

Van Hillegersberg and Kuldeep (2000) wrote in their article that Sherman C. Blumenthal proposed an integrated framework for organizational information systems already in 1969, although it was not realized until the late 1990s, when companies like SAP and Baan emerged in Europe. Since then several vendors have created software suites for ERP, but the market is still mostly dominated by SAP and Oracle. Oracle has also purchased other significant ERP software companies like JD Edwards and PeopleSoft during the 21st century to strengthen its position in the ERP market, but SAP is still the market leader holding a 28% market share. (Gartner, Inc., 2009)

Companies often expect ERP to solve all operations management problems, but it requires much more than just software to make a successful ERP implementation. Critical success factors have been identified in studies to be, for instance, business process re-engineering to fit the process models supported in the software, extensive training and top management support. Even though these would be taken into account, there are several other possible problem areas that can cause surprises during the implementation or use. Some examples are strong resistance to change, loss of key personnel and especially data quality degradation which is the focal point of this thesis. (Sumner, 1999)

2.2 LEAN SYSTEM ERP

Lean System ERP development was started in 1992 by a small university project spin-off company called Dialogos-Team Oy, which was later acquired as a part of Valtion Tietokonekeskus (VTKK) by Tieto Corporation Oy. Lean System relies currently on Oracle database version 10.2.0 and is mostly hosted on Microsoft Windows platforms. It is developed both with Unify Team Developer and Microsoft Visual Studio development platforms. (Raunio, 2010)

Lean System's main strengths are agility to adapt to several business models and production control methods simultaneously and the possibility to continue working with incomplete input data. It is based on the Pareto principle, as well as Agile and Lean production philosophies and mass customization to name a few. According to Lean System philosophy, the substance of resource planning is in user professionalism while the ERP visualizes the situation and provides tools to implement the required decisions. 80 percent of things can be automated but 20 percent of events must be brought to the professional user to evaluate and decide. This is supported by a robust user interface, alerting functionality, illustrative graphical displays showing the general picture and easy integration to other systems. Even though most events and circumstances can be fed to the system, all decisions and control operations cannot be predicted or automated. In addition Lean System has extranet functionality to support networked businesses and extensive production and project planning functionality with a graphical user interface. (Tieto Corporation Oyj, 2009; Tieto Finland Oy, 2010; Raunio, 2010)

2.3 DATA MINING

As mentioned earlier, DM is a combination of several techniques originating from different professions like machine learning, pattern recognition and statistics. It tries to solve problems by analyzing large data masses of data fast enough to provide meaningful and, in a sense, hidden knowledge. DM combines several techniques like sampling, estimation and hypothesis testing from statistics, in addition to search algorithms, modeling techniques and learning theories from artificial intelligence, pattern recognition and machine learning. It is also known to adopt new ideas from several sources like information theory, optimization, signal processing and visualization. Databases are often used to store and process the mining data if the data is not processed in its original location, which can also be a database. (Tan et al., 2006)

The commonly sought supplementary value from DM includes preventing fraud, giving marketing advice, seeking profitable customers, predicting sales and inventory and correcting data during bulk loading of the database, also known as the Extract-Transform-Load operation (ETL) (Lukawiecki, 2008). There are also very interesting scientific applications, for example, in the field of analyzing atmospheric data and recognizing certain diseases in the early stages (Fayyad et al., 1996a; Li et al., 2004). Moreover, in telecommunications it can also act as a layer in layered security architecture via malicious activity and outlier detection, which in part is one motivating factor for this thesis (Lee & Stolfo, 2000). Motivation for using DM comes from the value it gives over the competitors and it almost every time reduces costs, i.e. saves money, if the process is successful (Lukawiecki, 2008).

Two high-level DM goals are prediction and description. The first one tries to find patterns to predict the value or behavior of some entity in the future and the second one tries to find patterns describing the values or behavior in a form understandable to humans. These high-level goals are pursued through several DM methods; for example classification, regression, clustering, summarization, dependency modeling and change, as well as deviation detection. Each method has many algorithms that can be used to reach the goal, but some algorithms suit some problem areas better than others. (Fayyad et al., 1996c) These algorithms are discussed later in the chapter about DM algorithms.

The entities, i.e. data analyzed from data sets have attributes or features which differ according to the application area. Examples of different kind of data sets are discrete valued sequences, images, spatial data and time series. Data can be categorized also by its sparseness, structure and reliability to name a few possibilities. The attributes describing the data have also many properties pertaining to them; at least type, quality, preprocessing needs and relationships. An over-simplified example of an attribute is a field of a relational database table. (Fayyad et al., 1996a; Tan et al., 2006)

Attribute types are best illustrated by the next table by Tan et al. (2006) where nominal types are essentially names giving just enough information to separate two entities from each other, whereas ordinal types allow ordering of entities. Intervals and ratios extend types to differences between values and value ratios, like in calendar dates and densities.

Table 1 Attribute type classification. Source (Tan et al., 2006).

Attribute type	**Attribute subtype**
categorical (qualitative)	nominal
	ordinal
numeric (quantitative)	interval
	ratio

The important part of data quality issues can be classified under measurement and data collection errors and they have to be taken into account when planning DM activity. Noise and artifacts as well as precision, bias and accuracy problems can cause significant problems for DM process and sometimes their source is hidden from the DM analyst. (Tan et al., 2006) Consider, for example, a device measuring something at the production line having offset error of some kind.

Outliers, missing, inconsistent and duplicate values also need consideration of how to handle them in the source data. The second part of data quality issues fall under the application related problems and some general examples are timeliness, relevance and domain knowledge, i.e. knowledge about the data. (Tan et al., 2006) In fact this thesis tries to find a method to avoid some of these data quality problems via DM itself, but the concept of data quality relates to DM very closely so it had to be covered here for completeness.

There are several DM products available on the market, including DAT-A for MySQL, Oracle Enterprise Edition with Data Mining option, Microsoft SQL Server, SPSS Clementine, SAS and IBM Intelligent Miner for DB2 to mention a few. There are also Open Source projects for DM like Waikato Environment for Knowledge Analysis (WEKA) developed at University of Waikato (University of Waikato, 2010), Orange developed at University of Ljubljana, Slovenia (University of Ljubljana, 2010), and Togaware (Togaware Pty Ltd, 2010). This thesis is focused only on Oracle, since that database is easily available with Lean System ERP. The reader interested in more products is encouraged to visit the web site by Piatetsky-Shapiro (Piatetsky-Shapiro, 2009).

2.4 DATA QUALITY

The data quality discussed here differs from the data quality mentioned in the data mining chapter by trying to explain the importance of general data quality, not quality issues in the DM process. The problems are mostly the same.

Data quality is a key factor in an enterprise at operational, tactical and strategic level, since erroneous data causes wrong decisions and excess expenses. According to Redman, case studies show cost estimates from 8% to 12% of revenue merely at the operational level, incurred by poor data quality (Redman, 1998). When these errors proceed to the tactical and strategic level, they have even more wide-ranging ramifications. One monetary estimate of the losses is given in Eckerson's survey (2002) to be $600 billion a year just in US businesses, and this accumulates from only postage, printing and staffing. The other way around, data quality projects can give significant cost savings. This also emphasizes the importance of this thesis, since the results should serve as data quality control methods in addition to other methods like dependency checking.

A large portion of Eckerson's survey (2002) respondents (76%) also view data entry by employees as the main source of data quality problems, although it might be a subjective estimate, not a measured one. Other significant sources are said to be, for example, source systems changes, migration projects and user perception of the data, where the last refers to differing importance of the fields for separate users. This might cause them to fill fields with varying accuracy based on their importance to themselves, which can be easily illustrated by thinking of two separate departments. When the first department handles marketing and the second takes care of production, it is easily seen that production does not really care about customer properties and probably fills fields relating to them vaguely and, on the other hand, marketing does not care much about production measurements. (Eckerson, 2002)

Several attempts have been given to create a suitable list of attributes which characterize the quality of the data extensively. Newer comprehensive examples of these are written by Caro et al. (2008), who examine the data quality attributes for web portal data quality, and ISO/IEC standard 25012:2008 contains general data quality attributes, which the following table lists (International Standards Office, 2008).

Table 2 ISO/IEC 25012:2008 attributes for data quality (International Standards Office, 2008).

Characteristics	**Inherent data quality**	**System dependent data quality**
accuracy	X	
completeness	X	
consistency	X	
credibility	X	
currentness	X	
accessibility	X	X
compliance	X	X
confidentiality	X	X
efficiency	X	X
precision	X	X
traceability	X	X
understandability	X	X
availability		X
portability		X
recoverability		X

DM cannot probably affect all of these attributes, but a selection of them has to be chosen for measurement and suitable metrics have to be decided to evaluate DM process success. Chosen attributes and metrics for measuring data quality depend to a large degree on the requirements of the business (Eckerson, 2002).

Remarkable scientific research in this field is conducted at the moment by Massachusetts Institute of Technology's (MIT) Total Data Quality Management program which tries to establish a theoretical foundation for data quality research. The scope of their research is data quality definition, analysis and improvement. More information can be found from their website. (MIT, 2009)

2.5 DM APPLIED TO DATA QUALITY AND ERP

Artificial intelligence applications have been developed for many practical uses like speech recognition, computer vision and heuristic classification, even though the actual intelligence is not comparable to that of a human brain (McCarthy, 2007). Nevertheless, the application fields are already enormous, ranging from agriculture and natural resource management to video games and toys (Glick et al., 2008). Intelligent applications can also be seen as applications of artificial intelligence.

The viewpoint of several studies, like Eckerson (2002) and Orr (1998), seems to be that applications themselves do not have much functionality to enforce data quality, but they need an external method to maintain it. It appears to be that many software vendors have not adopted using application intelligence to improve data quality, either because of its complexity or just because they are not aware of such a method. Inputs are sometimes validated mechanically to include only allowed characters, but the validation would be much richer based on earlier values in the system. This means changing the validation to include application intelligence which could result in improved usability of software and give advantage over competitors in the software business, as already discussed in the overview of data quality and DM.

There are other approaches to data quality management like the Xiaosong et al. (2008) study of ERP data quality assessment and Information Product Mapping (IPMAP) by Shankaranarayan et al. (2003), but they have not yet been adopted widely either, probably due to their novelty and laborious nature. More approaches can be sought from (MIT, 2009).

DM is one of the possible external methods in maintaining data quality and also by definition artificial intelligence. It can be used for data quality management in addition to its common usage such as market basket analysis, but it is not normally an integrated part of software. This and other suitable methods for data quality management are listed in a study by Madnick et al. (2009). What strikes me as strange is that some companies do not seem to consider data quality as a continuous process but instead as a one-time setup, even though the opposite approaches do exist in practice. If it were a continuous process, it is reasonable to presume, that some controls and user training would be put in place to avoid future data quality degradation, but instead data quality is addressed solely when the used software suite is changed or some other major event. Some answers even reflect the attitude that the ERP system will somehow magically take care of the data quality problems which is not yet the case in today's software, as was noted by the researchers in (Xu et al., 2002).

This thesis tried to validate if an ERP system could integrate intelligent input validation with DM algorithms to help maintain good data quality in the absence of other data quality controls. Similar DM applications are commonly known as outlier detection and fraud detection (Lukawiecki, 2008). This DM activity resembles the classification method and suitable algorithms for this type of DM are discussed in Section 2.6, but other methods might be possible to use. One notable risk in this approach is that it needs at least some amount of good quality data in the database for training the DM algorithms and constantly fed bad quality data will eventually train the algorithm to think that this data is as it should be. This implies the need for at least a one-time check for the data quality before training the model to avoid creating a model not noticing any real anomalies.

2.6 DM METHODS AND ALGORITHMS

There are many methods in DM and suitable ones available in this case are classification, clustering and regression analysis. Some of these methods are supervised needing a target attribute or known value, while others are unsupervised and do not need a target and known values. (Oracle, 2008b) This case can use both types of methods; just the application procedure differs between the two and both can help in finding the allowed values.

The classification method tries to categorize items to predefined target classes with the help of some algorithm. The building of classification model includes training data set with known, discrete target classes, which means that the classification results are always discrete. Classification targets vary from binary to multiclass attributes and the models try to predict which target class is correct with the help of descriptive relationships from the input attributes. Classification has numerous algorithms publicly available with varying application targets, from which some examples are Decision Tree, Bayesian networks, Support Vector Machines (SVM) and Rule Induction. (Oracle, 2008b; Maimon & Rokach, 2005)

The clustering method builds clusters based on some similarity measure like density or distance. At the moment there are at least four main types of clustering algorithms, namely partitioning, hierarchical, locality-based and grid-based. Partitioning puts the cases to clusters based on their nearness, which is usually defined by distance from the cluster center defined by mean or choice. This is probably the most dominating clustering type. Hierarchical clustering builds tree structures from top to bottom or vice versa by dividing or combining cases based on the chosen distance criterion. Locality-based clustering, on the other hand, uses local conditions to build clusters, for example, diminishing density near some members of a cluster can be interpreted as a valid edge for a cluster. Finally, grid-based clustering divides the input values to hyper-rectangular cells and combines continuous cells to form clusters. (Oracle, 2008b; Milenova & Campos, 2002) Common algorithms are, for example. K-Means, Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH), Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and OptiGrid (Zhang et al., 1996; Ester et al., 1996; Hinneburg & Keim, 1999; McQueen, 1967).

Regression methods resemble classification, but the output is commonly a numerical estimator for a continuous attribute, not a class label. Regression builds a function of independent attributes and that function attempts to predict a dependent attribute. Classical regression has several assumptions, like target attribute being normally distributed and constant variance, but these have been improved in new algorithms. The first regression algorithm is known as the least squares method and it was published a long time ago in 1805 by Legendre. After that there have been published several improved versions like SVM (Vapnik, 2000) and GLM (Nelder & Wedderburn, 1972) and most of them still use the least squares method in some form. (Legendre, 1805; Oracle, 2008b)

As seen, all methods can be implemented with several algorithms. One requirement for these algorithms is that they will preferably be run inside the same system where the original data resides, since separate software would require more licenses and cause more expenses to clients. According to Oracle, this approach has also several other benefits like no data movement is necessary, better security, easier data preparation and data freshness (Oracle, 2008b). The initial criteria for the algorithms to be used in this work are presented in the next table and they were relaxed at later stages.

Table 3 Algorithm criteria and explanation.

Criteria	**Explanation**
Included in the product	No self-coding possibility due to resources.
Scalability	Needs to scale well from small data sets to large data sets.
No dimensionality curse	Works with high dimensionality data i.e. large amount of attributes.
Rarely trained	Requires new training rarely.
Robust against over-fitting	Reduces chances of fitting noise.
Easy deployment	Easy to use as part of SQL clauses.

Oracle DM Concepts manual seems to use a different vocabulary about some methods and does not suggest, for example, that clustering is suitable for anomaly detection, but based on other sources like Hodge & Austin (2004) and Lukawiecki (2009b), the following built-in algorithms might be usable (Oracle, 2008b).

Table 4 Algorithm availability in Oracle (Oracle, 2008b).

Algorithm	**Method**	**Initial thoughts**
K-Means (KM)	Clustering	Might not scale well for large data sets; requires training for every data set; order of data changes behavior and random starting point; complexity O(n).
One-Class Support Vector Machine (SVM)	Classification, regression, anomaly detection	Scales well; computationally complex and might be difficult to train.
Orthogonal Partitioning Clustering (O-cluster)	Clustering	Scales well for large high dimensional data sets.
Minimum Description Length (MDL)	Feature selection	Could be used for feature selection verification before actual data mining.
Generalized Linear Models (GLM)	Classification, regression	Scales well; requires certain distribution from data.
Naïve Bayes	Classification	Scales well; fast.

The K-Means term was coined by James McQueen in his paper studying classification of multivariate observations in 1967 and the same algorithm was also proposed earlier by Stuart Lloyd in Bell laboratories in 1957 for pulse code modulation (PCM) (McQueen, 1967). The latter was missing some numerical calculations and was not therefore published before 1982, so the original credits went to McQueen, while he also apparently based his studies on some earlier incomplete works.

K-Means tries to partition entity space to *k* number of groups based on their distance from *k* random calculated means. When an entity is placed in a group, the mean is recalculated and the round begins again with a new sample. If some groups are very close to each other, they might be combined to form a new single group with a single mean and distant entries can also form new groups. There is, however, no guarantee of an optimal solution and the formed clusters also differ on each run due to random starting points. The algorithm has several applications from color reduction to pattern recognition and has endured as a practical approach ever since. (McQueen, 1967; Lloyd, 1982) The following figure shows a K-Means clustering example, where the red cluster represents outliers and two others are dominating clusters.

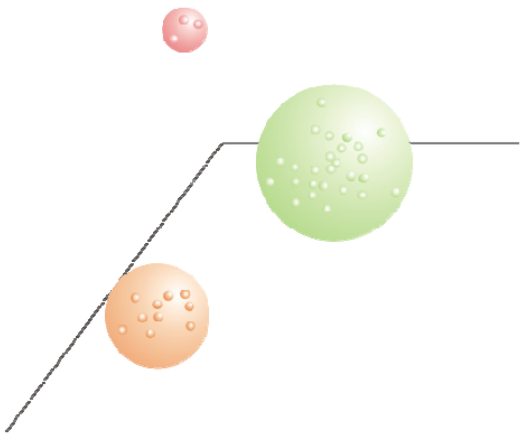


Fig 1 Three clusters in three-dimensional space.

O-clustering is Oracle's proprietary algorithm, published by Milenova and Campos in 2002 and based on an algorithm called OptiGrid. The basic difference between the KM and O-cluster is that the latter is so called grid-based clustering while the KM uses partitioning by distance. The O-cluster builds axis-parallel grid and bases its decision on continuous density concentrations, instead of distance and performs extremely well when the dimensionality increases. In comparison, partitioning suffers when the dimensionality grows, since the distance differences become almost zero due to the so-called nearest neighbor problem reported by Beyer et al. The KM also builds as many clusters as there are given as a parameter and the O-cluster builds only as many clusters as it can really find, given the boundaries by other parameters. An illustration of the O-clustering grid is given below. (Milenova & Campos, 2002; Beyer et al., 1999; Hinneburg & Keim, 1999)

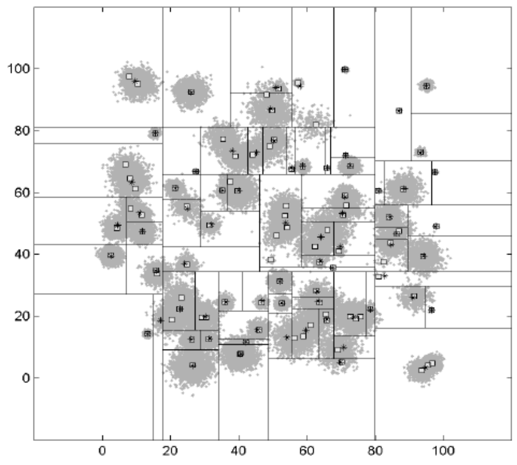


Fig 2 O-clustering example. Source Milenova & Campos (2002).

Classification has the most extensive set of algorithms available in Oracle, namely Naïve Bayes, Decision Tree, GLM and SVM. The Naïve Bayes calculates the conditional probability of a value belonging to a class from Bayes' Theorem. It does this by counting the combinatorial probability of past values for some source attributes and a target attribute, then scaling that with the probability of the source attributes. It makes an assumption about the independence of the source attributes and everything belonging to just one class, causing it to overlook any dependencies, hence the name "naïve". It was first used for classifying purposes by Maron in 1961 and it is still an extremely popular algorithm due to its simplicity. (Maron, 1961; Maron & Kuhns, 1960)

SVM was also introduced a long time ago, in 1963, as a linear classifier. In 1992 it was improved with a so called kernel trick, which enabled it also to act as a non-linear classifier by using a non-linear kernel. This makes it possible to use linear classifiers in non-linear cases to model quite complex distributions. The classifying is done by constructing hyperplanes which separate the classes in a high dimensional space, trying to keep the margins as large as possible between the classes. SVM can also be used for regression by trying to fit the samples inside some margin from a function approximating the samples. It has been improved several times after its invention and according to Milenova, they have optimized Oracle's SVM algorithm further to avoid problems with scalability, model size and performance to gain better adoption. (Vapnik, 2000; Milenova et al., 2005; Boser et al., 1992; Smola & Schölkopf, 2004) An example of a hyperplane separating two groups of samples is represented below.

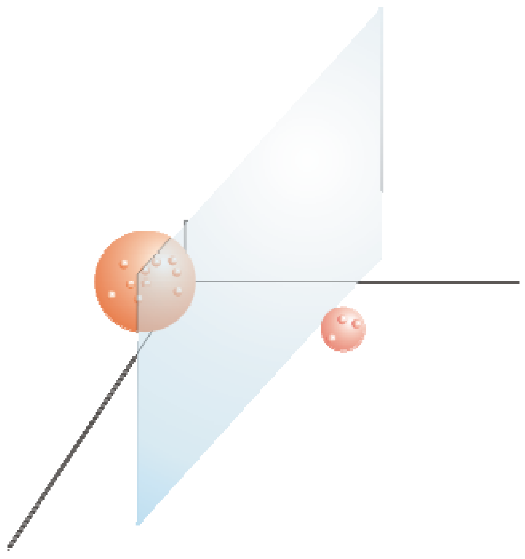


Fig 3 SVM hyperplane separating two classes.

GLM is an improved, generalized version of linear regression analysis; overcoming some restrictions like need for constant variance for the target attribute and supporting other than normal distributions through a link function. It was published by Nelder and Wedderburn in 1972, so it is also well tried and tested. Regression analysis tries to fit a function to a data set by finding suitable parameter values to accompany the independent predictor attributes. Together these parameters and predictors try to model some dependent continuous numerical target attribute, while some error is allowed in the prediction. (Nelder & Wedderburn, 1972) The next figure contains an example of regression where the function curve tries to predict the values of *y* based on the values of *x*.

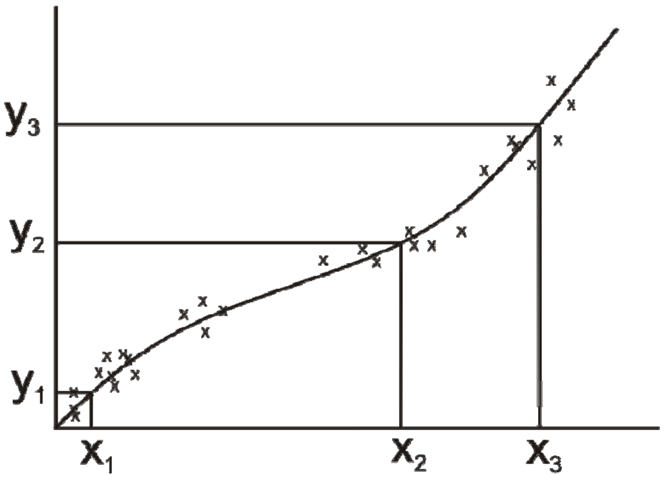
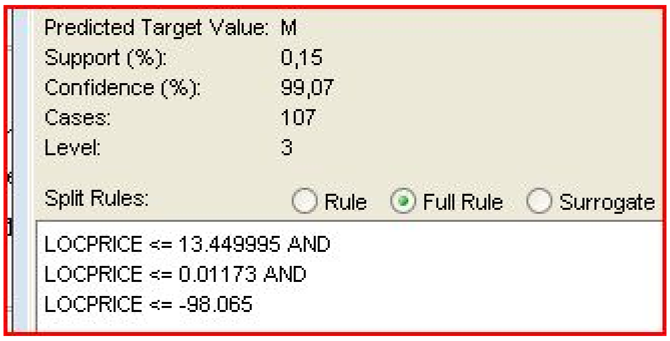


Fig 4 Regression line where *x*'s are values of an independent predictor attribute and *y*'s are predicted values of a dependent attribute.

The Decision Tree algorithm resembles Naïve Bayes, because it is based on conditional probabilities, but in addition it generates simple, user intelligible rules. It is very descriptive in explaining what attribute values cause something to happen, but requires good descriptive input attributes to succeed in explaining. MDL was published by Rissanen in 1998 and has been the subject of continuing studies ever since. It is based on an assumption that the most explanatory input attribute for a target attribute is the one maximizing the compression of the target data. In Oracle this can be applied to attribute selection by analyzing which of the available input attributes are the most descriptive for a target attribute, i.e. gives the best compression. (Breiman et al., 1984; Grünwald, 2004) The following figure is a capture from an actual Decision Tree experiment during this thesis, where the rule conditions are seen clearly.



```
Predicted Target Value: M
Support (%):            0,15
Confidence (%):         99,07
Cases:                  107
Level:                  3

Split Rules:            ○ Rule   ● Full Rule   ○ Surrogate

LOCPRICE <= 13.449995 AND
LOCPRICE <= 0.01173 AND
LOCPRICE <= -98.065
```

Fig 5 Decision Tree sample rule.

Since the suitability of different algorithms is not clear before testing them empirically on chosen data sets, the actual choice of algorithm was left to the practical part of work, even though domain knowledge might help a more experienced DM practitioner to choose an algorithm beforehand. Let it also be known that most of the DM algorithms are publicly available and, therefore, the ready-made algorithm selection can be expanded. The problem with expanding the selection is that software vendors tend to make changes to these public algorithms and optimize them for their products, as can be seen with Oracle's proprietary versions. The level of maturity in these proprietary versions can be quite difficult to achieve, if one does not have extensive knowledge about mathematics behind the algorithms.

2.7 DM AS A PROCESS

The purpose of this section is to show that there are structured ways to go through a DM process to minimize a possibility of failure and wrong decisions (Fayyad et al., 1996c). These processes are referred to in literature as processes, methodologies or frameworks, but a common goal for them is to make sure that the end result is useful, even though the level of abstraction of these processes vary considerably. Several formalized process models have been proposed since Brachman and Anand's study of knowledge discovery process and their level of abstraction varies considerably (Brachman & Anand, 1994).

The first actual process model is said to have been created by Fayyad et al. during the mid-1990's and after that, for example, Cabena et al., Anand & Buchner and Anand et al. have all tried to create new models for different purposes according to Kurgan & Musilek (2006). My intention is to use a process model called "Cross-Industry Standard Process for Data Mining 1.0" (CRISP-DM) which was also introduced in 1996 by a consortium of four companies: commercial DM software manufacturer SPSS, database provider NCR, car manufacturer Daimler Chrysler and an insurance company called OHRA. This process model is widely used in industry and supported by a special interest group of several hundred DM users and tool vendors which makes it very practical. It is also the basis for Oracle's data mining framework and much of the literature for DM refers to its graph about the DM process. (Chapman et al., 1999; Kurgan & Musilek, 2006; Oracle, 2008b)

The actual process is not strictly ordered but more like an iterative process repeated over and over again, probably returning to the earlier process steps every now and then when a need arises. The arrows in the next figure signify the possible iteration paths or feedback loops. The intention is to create most of the outputs required by the CRISP-DM process model to facilitate an easier DM application and to verify that all necessary steps are performed. Other process models would probably work as well.

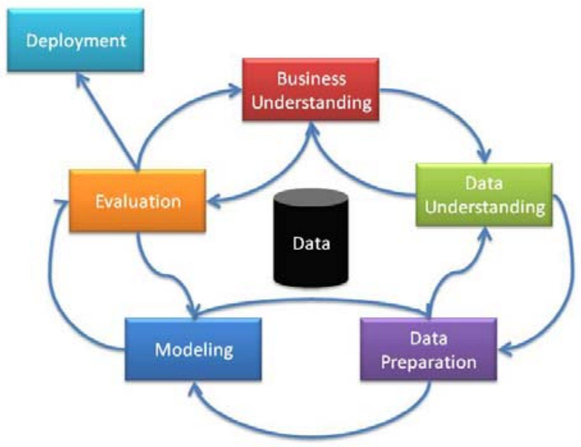


Fig 6 CRISP-DM process circle. Adapted from (Chapman et al., 1999).

2.7.1 PROBLEM IDENTIFICATION

Identifying the problem is the first task in the CRISP-DM process. Understanding the correct business problem, i.e. business objective to solve, can be very tedious and difficult, but it is nonetheless very important. Defining the right problem might as well be possible by breaking a larger problem into small sub-problems and solving them independently, but asking the right questions is often the hardest part with every problem. Misunderstanding the problem might produce useless results and might make the whole process fail, even if the algorithms chosen produce accurate results. This can lead to false decisions which in turn can be very costly. Sometimes even exploring the problem space gives insight into the solutions needed. (Pyle, 1999; Chapman et al., 1999)

Cognitive maps and pair-wise ranking with ambiguity resolution are very effective methods in achieving a precise problem definition, ranking problems and communicating them to the actual data miner. Cognitive maps are graphs which display the concepts and links between them, forming the problem space. Links represent the positive or negative interactions between each concept and they can be uni- or bidirectional. These maps give insight into the different viewpoints that can be brought to the problem. A good example is in the following figure, where the same concept has two completely different viewpoints, depending on the amount of variables introduced. (Pyle, 1999)

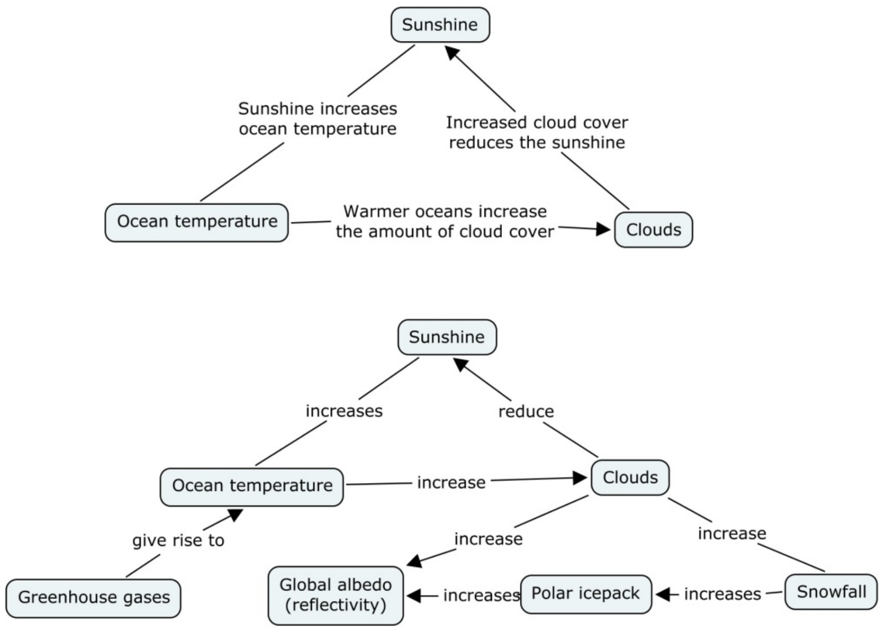


Fig 7 An identical concept with two completely different viewpoints. Adapted from (Pyle, 1999).

Pair-wise ranking helps in identifying the most important problems when several are available and the order of significance is not clear. As an example, the problems are compared one by one for importance, difficulty and yield, then each of these properties are weighted and a weighted problem importance calculated. A modern spreadsheet or decision support system can aid in doing these calculations. Ambiguity resolution ensures that if there is a tie between two problems, one of them will be chosen as the most important one. In addition, it is extremely important that the assumptions associated with each problem are communicated as clearly as possible to the DM practitioner to avoid difficulties during the process. (Pyle, 1999)

In a DM process a solution has to be defined as well. The data miner has to know what is being sought as an outcome to know when a solution has been reached. The output has to be specified very clearly, since it could be, for example, a report, chart, graph or algebraic formulae to name a few possibilities. Output specification has to be also practical enough to be implemented, but it can evolve during the process. (Pyle, 1999) The ideal target of this thesis is to find an algorithm, which does not require manual intervention during training, to be implemented as an automatic alerting system against false data.

The next part in the first step is the actual implementation specification to gain the business benefit the whole process was started for. A practical definition of the implementation might be difficult to form, but without it, the whole process cannot be completed. Practicality means that the specification has to state the problem it solves, the form of the solution, the provided value, for whom it has been done and how, as well as restrictions etc. to make sure that the solution can be implemented successfully. For evaluation purposes a success criteria for data mining should be defined as well. (Chapman et al., 1999) As discussed by Pyle (1999), it has been said that this task takes 1 percent of the time but is 51 percent of the success factor.

2.7.2 DATA UNDERSTANDING

The second step of the CRISP-DM model concentrates on the data understanding. This requires that the DM practitioner collects the data, explores it to find if any subsets of the data are particularly interesting, identifies data preparation needs and recognizes data quality problems. These data quality problems are discussed in two phases: general data quality issues mentioned in Table 2 and issues directly related to DM activity, which are listed in the following table. (Tan et al., 2006; Chapman et al., 1999)

Table 5 Data quality problems directly related to DM activity. (Tan et al., 2006)

Characteristics	**Explanation or example**
measurement errors	deviation of measurement from the true value
collection errors	e.g. all values not collected for some reason
noise	random error component in measurement
artifacts	caused by deterministic phenomenon like compression
precision	closeness of several measurements of the same quantity or rounding error
bias	systematic variation in measurement process
accuracy	degree of closeness of a measurement to its true value
outliers	value numerically distant from the rest of the data

Characteristics	Explanation or example
missing values	empty values where they should be present
duplicates	duplicate values where uniqueness is required
timeliness	data is current, i.e. from relevant time
inconsistency	data inconsistencies, e.g. wrong zip code for a certain city
relevance	data contains information necessary for the application
dimensionality	referred often as curse of dimensionality; sparseness of data caused by too many dimensions

Humans can introduce some of these problems like measurement, collection, missing value, duplicate and inconsistency errors when a user inputs the wrong values and this is the exact problem this thesis is trying to solve. Other sources for these errors are, for instance, device limitations or faults and these are very probable to be solved with the same kind of means as human errors. (Tan et al., 2006)

2.7.3 DATA PREPARATION AND ATTRIBUTE SELECTION

The third step of CRISP-DM is data preparation, which contains several tasks in a non-predetermined order, like, for example, transformations, views, transfers, loads, aggregation, sampling, dimensionality reduction, attribute selection, and attribute creation, discretization and binarization – if needed. Its main significance is to prepare the raw data before feeding it to the DM algorithm to make it more suitable for mining and to handle the data quality problems recognized in the previous step. The next table explains the common procedures used in data preparation. (Tan et al., 2006)

Table 6 Procedures used in data preparation. (Tan et al., 2006)

Preparation procedure	Explanation
transformation	transformation by simple functions like summation or normalization (standardization)
outlier handling	extreme values might have to be removed or replaced with something closer to distribution

Preparation procedure	Explanation
missing values handling	missing values might have to be replaced with something more meaningful
binning	places cases in histogram-like buckets
transferring	transferring data to other location before mining
loading	loading data into mining platform like database
aggregation	combining several objects to a single object, e.g. summarizing
sampling	taking a representative sample of the whole data set to facilitate easier or faster processing
discretization	categorization of a continuous attribute
binarization	transforming continuous or discrete attributes to binary attributes
dimensionality reduction through attribute selection or creation	subset selection of attributes present in source data or creation of new attributes from attributes present in source data - see following paragraph

Preprocessing steps needed in this case are probably outlier and missing values handling, normalization with skewed attributes, binning, discretization, binarization and sampling with large tables. Some of these can be handled automatically by the database interface to the DM functionality, but manual transformation might be required in some cases. Outlier, missing values handling and normalization are related to data quality correction and several algorithms require these always as a prerequisite. The details of these transformations are discussed more thoroughly in Section 3.5.

Attribute selection, also known as feature selection and is a part of data preparation, is a crucial component of the DM process and is a research field of its own. In short, it signifies which source data set attributes are selected as an input to the algorithm. It can be done using domain knowledge, weighting and ranking attributes with filter and wrapper models or by selecting all available attributes. An ideal solution would be to test all available combinations of attributes and compare the accuracy of each selection, but high dimensionality will make this infeasible quite quickly since exhaustive search would have to go through 2^n combinations when there are *n* attributes. There is also a possibility of over-fitting the model, which means that the DM model generated will model also the noise and errors thus having poor predictive performance when applied. (Tan et al., 2006; Jensen & Cohen, 2000; Liu & Yu, 2005)

A filter approach evaluates attributes through some heuristics applied to the actual data or attributes, while the wrapper uses an actual learning algorithm like MDL to evaluate the usefulness of the attributes. There are also hybrid approaches using both. Their common goal is to reduce dimensionality, remove noise and to improve mining performance through the speed, accuracy, simplicity and comprehensibility of results. One other mechanism for attribute selection is to create new attributes from old ones through some algorithm. This is often called feature extraction and, in practice, it means feature reduction through mapping from high-dimensional data to a low-dimension version. This tries to preserve the amount of information compared to feature selection using only a subset. (Yu et al., 2007)

For the test cases in this thesis, domain knowledge was used from Lean System product development team, while there are frameworks like the one proposed by Yu & Liu (2004), intended to help in attribute selection. They could be used in later stages to formalize attribute selection in addition to algorithms built for the same purpose. Attribute selection in this case had to be made without much knowledge of the data to be analyzed since data is different in each target database depending on the use of the ERP system. Attributes can, in any case, be selected manually with domain knowledge to be relevant and non-redundant by following the definitions in the work by Liu & Yu (2005), but the result is not necessarily an optimal subset of attributes, just an approximation as it is even with the previously mentioned framework. There is also some built-in attribute weighting in some algorithms like Oracle's SVM (Oracle, 2008b), but pre-choosing attributes gives apparently better results. Another possible approach is to use a wrapper model running the algorithm in use to identify the suboptimal subset of attributes (Liu & Yu, 2005).

In addition to the accuracy, there are several measurement scales for the optimality of the subset selection. For example, information measures with entropy and information gain, consistency measures with consistent class separation with minimum set of attributes, distance and dependence measures based on partition distance and attribute dependence can be used; however the main concern of this work is the accuracy of the prediction. (Yu et al., 2007)

2.7.4 MODELING

The fourth step in the CRISP-DM process is modeling by the actual DM method. This step has several sub-steps from which the first is choosing the method from at least classification, regression, clustering, summarization, dependency modeling and change or deviation detection. The next sub-step is to implement the chosen method with some algorithm from the previously mentioned list of available algorithms in Oracle. This requires verifying the assumptions the algorithm has against the data quality and format and probably causes a reason to revisit the preparation step, if the assumptions are not met. (Fayyad et al., 1996c; Chapman et al., 1999)

This step requires also compiling tests to verify model quality and validity for the purpose in hand. In practice this means separating some data as a training set and estimating its quality on a separate test set, where the measure of quality is, for example, an error rate on the testing set. Training and testing data sets can be from the same data source divided by some percentage and randomly selected, so that both sets contain timely data. Oracle's product can do this also by itself for certain algorithms, if the DM practitioner has no need to create tests manually. Common percentages seem to be from 60% to 80% for a training set from the complete data and the rest of it for testing, i.e. from 40% to 20%. Some approaches suggest that partitioning the data into several buckets and cross-checking the remaining buckets with each bucket gives insight into the amount of data needed for training and accuracy of the model. The earlier warning in Section 2.7.3 about overtraining is valid, so too much training is not advised. If present, accuracy estimators are different for each algorithm and the results vary depending on parameters, training data and testing data. (Oracle, 2008b; Chapman et al., 1999)

One notable problem with training is the availability of clean data. If the training has to be done with the data available in the customer production database, it very probably already contains anomalies and outliers which affect the model outcome. Outlier tolerating algorithms are called robust algorithms and at least Oracle's SVM has been improved to tolerate some percentage of outliers in training data (Milenova et al., 2005). One other possibility would be to clean the data before training the model, but this might be too laborious step to go through with every data set. Therefore, training will probably be done with whatever data there is in the database, with the exception of running the required missing value, outlier and normalization procedures. (Tan et al., 2006)

The next sub-step is building the model, which requires that the parameters of each algorithm are adjusted and tested to give the best possible accuracy. This also reflects the iterative approach necessary to DM since testing and re-adjusting is probably needed. After this follows assessment which effectually means comparing the success criteria and test criteria defined earlier in the process to the actual results. Algorithms can be ranked at this stage to enable choosing the correct one for final DM, based on, for example, their accuracy, success, reliability and deployment potential. (Chapman et al., 1999)

2.7.5 EVALUATION AND DEPLOYMENT

The fifth step in the DM process tries to evaluate how well the business objectives are met with the chosen algorithm or if there is some reason the chosen approach can be declared faulty. If external constraints like budget and time allow, the models can be tested with separate data from the real application which fits the testing data definitions. The CRISP-DM process model also instructs to check additional things such as whether the results are understandable, interpretable and useful and if they have raised new business objects for some reason. Evaluation also includes a review phase to make sure the process has been implemented correctly and there are no forgotten steps or other errors, which can trigger a new iteration of the process. (Chapman et al., 1999)

The final sixth step is deployment which concentrates on creating the actual deployment into business use. This is achieved by planning the steps necessary to implement a solution including end-user information delivery, measurements for benefits, measurements for applicability, monitoring the functionality of the solution over time, software implementation and organizational challenges. Review of the whole process and final reporting are also necessary phases for learning from the whole DM process and might also be required by the business owners. (Chapman et al., 1999)

2.8 DM IN ORACLE

Oracle enables its DM functionality through interfaces which are the Oracle Data Miner graphical user interface (ODM GUI) in addition to Procedural Language/Structured Query Language (PL/SQL) and Java Application Programming Interfaces (API) (Oracle, 2008b). ODM and Java API are both layered on top of PL/SQL which implies that an integrated solution should use native PL/SQL interface if possible to avoid speed degradation and redundant layers.

PL/SQL exposes DM functionality through three packages and normal Structured Query Language (SQL) clauses can include DM models as a part of their structure, which in turn enables easy integration into SQL-enabled applications (Oracle, 2008a). This is probably the subject of future research in this case, but nonetheless one requirement for this whole work. In this thesis the DM activity was performed through ODM GUI to facilitate easy preprocessing. The models created through the GUI are also implementable in PL/SQL through automatic package creation and can be easily transferred to ERP system code.

Oracle has optional Automatic Data Preparation (ADP) functionality to perform the needed preprocessing steps for each algorithm. This reduces a significant amount of work from the DM process, if used. Possible transformations depend on the algorithm, but commonly available ones are normalization, binning, outlier handling and missing values handling. Algorithms also have some automatic handling for data quality problems like missing values by replacing them with mode or mean, depending on the data type. ODM also enables the user to build views, transform and summarize source data sets through a wizard-driven GUI. (Oracle, 2008b)

The Oracle DM approach to the DM process is similar to the CRISP-DM model with some differences in naming the steps, changing the order of the steps and combining steps. This does not affect the outcome, since the whole process is iterative and all steps are gone through anyway with all the necessary steps included. (Oracle, 2008b; Chapman et al., 1999)

3 TEST PREPARATION AND REQUIREMENTS

This chapter discusses the business and data requirements, the preparation for the DM process, problems encountered in preparation, resources and the source data set used during the research.

3.1 RESEARCH QUESTION BOUNDARY CONDITIONS REVISITED

All objectives and chosen approaches are presented again for clarity in the following table.

Table 7 Objectives for the DM process in this thesis.

Objective
No hard-coded rules for anomaly detection.
Only built-in algorithms from Oracle at this point.
Scalable and high dimensionality supporting algorithm.
Alerting the user when the submitted information does not conform to previously submitted information causing probably failure, i.e. missing or erroneous information present.
Data quality improvement by DM.
CRISP-DM process model.
Training with some percentage of current data, no clean up necessary.
Testing with some percentage of current data.
Easy implementation and usage as part of SQL clauses.

The most significant risk in this study was related to the author's low statistical skills and algorithm expertise, which both can cause false conclusions in the preparation phase, modeling and result interpretation. One other lower level risk was bad training data quality and even though it can be solved with cleaning, that can be very expensive and arduous task in large masses of data. The most practical risk is dimensionality curse, which rose to be a significant part in some algorithms.

The sample data sets were two tables from Lean System ERP, suggested by Lean System project manager and the attribute selection was done with domain knowledge. I assumed the needed preparations would be at least normalization, discretization, binarization and sampling, depending on the algorithm.

3.2 TEST SETUP

The needed software for the research was Oracle Database Server 11g1 Enterprise Edition with Data Mining Option, even though Lean System's database platform is currently version 10g2. Future versions will probably be released under 11g2, so a newer version was justified. Oracle SQLDeveloper 2.1 development and editing front-end was also installed to facilitate easier querying and developing of SQL clauses.

These were installed on top of a Windows Vista SP1 platform for testing purposes. In a normal operating environment, the Oracle database software could be installed on any supported platform, including Microsoft Server operating systems, Linux derivatives and HP-UX to name a few. The hardware running Windows Vista was a Fujitsu Lifebook P8010 laptop with 2GB of physical memory and a standard 146GB hard disk with encryption enabled, so the computing power needed for modeling is not so different from a standard PC in this case. Larger data masses could need more computing power, but the basic idea behind DM is that even though the modeling might take some time, the actual application to the target data is fast.

Data sets were exported from the source databases with Oracle's export utility and imported into the local database with Oracle's import utility. The database was created with similar settings as normal Lean System customer databases, except for the Oracle instance 512MB memory allocation. This is quite low compared to production databases with memory allocations of 3GB or more. A laptop having only a single disk also restricts the disk subsystem performance a great deal compared to production systems and encryption on top of that caused even more stress.

ODM required extensive rights to the database objects for the mining user. At first an attempt was made to create a minimal rights user based on Oracle installation instructions, but this approach failed in getting all the information from the database administration views, so the end result was using a mining user having the DBA role for the whole database. In production systems this is not a viable approach due to security problems in giving too broad rights to users.

3.3 BUSINESS UNDERSTANDING

As previously mentioned, data quality problems cause costs through bad decisions and correcting errors caused by bad data. Consequently, a business objective was selected for CRISP-DM to be saving money by improving data quality, which in practice turns out to be warning about possible erroneous inputs. In this project, the target was still just theoretical to verify if this is possible at all. As a possible continuum, it was necessary to find out which algorithms seem to be good candidates for this kind of usage and how could they be implemented as a process.

A business success criterion was selected to find an algorithm, or several algorithms, to spot outliers with DM when new data is fed. This will cause cost savings as earlier studies reveal, but estimating monetary benefit was too laborious according to domain experts in Tieto / Lean System, since every error and every customer have different kinds of costs related to errors. Instead they rely on the fact that some customers have had costs incurred by these kinds of problems earlier and they have been repeated and are large enough to justify this kind of study about DM applicability. Finding good cost averages would, in the author's opinion, probably require a large quantitative study in the future.

The current solution for input checking is quite simple and ineffective against logical errors. – An input check for valid characters verifies non-numerical entries in numerical fields or erroneous characters in inputs. It is odd that hard-coded rules seem to be overlooked, since they would probably also work well for checking constrained attributes, for example in cases where the input value deviates more than an order of magnitude or where it is physically impossible. Based on this, statistical methods should not be overlooked in input verification.

Making an inventory of the resources available for this project was easy since the only available person was the author, if not counting consulting given in interviews with Lean System domain experts; hardware resources have been mentioned earlier. It was anyway beneficial to find as many sources of information as possible with domain knowledge of either DM or ERP to reflect on ideas and thoughts of the author throughout the project. Data sets needed were provided by actual old operational data to reflect a real-life situation.

Requirements were quite normal for this kind of project, except for the relaxed schedule, which was possible even though money is involved in developing software. This was possible because this is a thesis work instead of a customer project and research of future trends is nevertheless seen as important. The comprehensibility of results for internal as well as external readers is an implicit requirement for the thesis but still mentioned here because the CRISP-DM suggests identifying comprehensibility requirements. An easy deployment plan for implementing results was also needed to ease the starting of possible future work by someone else and maintainability for keeping models fresh with minimal work can also be seen as part of that. Finally, repeatability is a target and needed if DM will be used as a part of the product in the future.

In the middle of the project, domain experts from Tieto posed a question about the nature of the data in Lean System ERP. They wanted to know if DM would reveal aspects of data which could be checked with hard-coded rules and this became an additional requirement for the project.

There were also three important requirements for legal, security and price aspects. The test data had to be acquired from real-life operational data with permission from customers. Security had to be maintained for that data and it had to be kept anonymous by removing all customer specific knowledge from the analysis. The price aspect required that the license price compared to benefits should be low enough to attract customer attention which means it should not require additional investments in new Oracle licenses. In practice, this could be impossible if Oracle's own ODM is used but with Open Source alternatives it might prove possible to create similar functionality with development costs. On the other hand these development costs can rise to the same level as the ODM license, because developing mature and easy to use DM algorithms can prove to be time-consuming.

Approximations of license fees and future development work for implementation are presented in the next table to help in understanding the costs related to this kind of functionality. A normal Lean System ERP database license is Oracle Standard Edition One, which means that Oracle Enterprise Edition and an additional Data Mining license need to be purchased to gain functionality for ODM.

Table 8 Approximation of license fees and development work for DM functionality in ERP system.

Subject	**License**	**Other information**	**Price**
Oracle Enterprise Edition (EE) w/ODM	10/1/2009: EE named user + support or processor license + support. DM option named user + support or processor + support.	Depends on user amount, often at least 20 named licenses.	Single user price at least eight (8) times the price of standard edition.
Oracle Standard Edition (SE) One if Open Source algorithms used	10/1/2009: SE One named user + support or processor license + support.	Depends on user amount, often at least 15 named licenses.	Open source algorithm development costs.
Costs for data collection in implementation	n/a	DM done in the source database.	0
Training time for users	n/a	Depends on the implementation and user abilities.	Trainer price + lost work time per person.

Subject	License	Other information	Price
Implementation costs for cleaning, training, testing and installing feature	To be decided.	Preprocessing probably very laborious. Experience with different distributions helps later.	Not known.
Costs of developing the application to use DM	Not known.	Application changes simple, algorithm training and data preprocessing laborious.	Not known.

Certain assumptions were made during this work based on general description of DM capabilities and data set quality. It was clear that DM algorithms can find outliers, but it was not known how it can be done with real-life ERP data sets and what problems will raise. Training data will never be completely clean of errors, which has to be taken into account in algorithm training. Real-life data availability would also not be a problem since some customers are very interested in research. In addition, results from modeling would need interpretation since they are based on probabilities, not certainties. The final assumption was that even if DM functionality in input checking would be laborious to implement, it would give competitive advantage against other ERP manufacturers.

The DM goal was defined to be alerting users about possible erroneous input. Creating a database trigger or application code to do the actual warning is close to trivial, but getting the DM algorithm to find outliers is not and it was not known which algorithms would support the data sets selected as targets. Probable methods were initially thought to be classification, clustering and regression. Accuracy was not known and other DM applications had had percentages of 2%-97% of known outliers found, so the accuracy distribution is quite wide (Milenova et al., 2005). This is due to the fact that before knowing the source data set distribution, descriptive attributes and outlier types, it is impossible to give any estimates and even then modeling can produce surprises.

In practice the business goal and success criterion were fulfilled by testing all ODM algorithms against the chosen attributes. This brought out the suitable algorithms, answered the research question and gave lots of information about the distribution of customer data in target tables. Some ideas for implementation methods were also gained.

3.4 DATA UNDERSTANDING AND ATTRIBUTE SELECTION

All the data was already in the Oracle database, so no data transfer was originally necessary. Data sets collected were one version database from Lean System version 6.0 and one customer operational database copy. The version database is a database used for testing and installing a certain version of Lean System. The customer database copy was from an older Lean System 5.3 version but selected target attributes had not changed between these versions. More customer data would have enabled more experiments with different distributions of target attributes, but the data used was enough to verify the hypothesis about input validation. The target tables were selected based on Lean System expert interviews to be tables containing stock and purchase invoice details.

Attribute selection is a very important part of the DM process and it took a significant amount of time. The selected source data sets had 154 and 183 attributes in stock details and invoices respectively. Some attributes were clearly not needed and easily discarded but some had to be eliminated or selected with domain knowledge provided by Lean System's system architects. As a prerequisite, ODM requires that source data is in one table or view, with the exception that some algorithms support nested tables and, in addition, each case has to be in one row of data.

In practice the attribute selection was simplest to do with views selecting as few attributes as possible from the source data. If almost the same set of attributes is used for several mining models, they all can be included in one view and only attributes needed by a model are selected at the time of model creation. Constant and simple naming models are good to adopt at this stage for the views and other database objects, since there are probably numerous names to be generated during the process. Good naming helps in distinguishing the objects directly from the name.

The initial analysis of target tables was performed through a specialized SQL query. The query was run in SQLDeveloper and results were exported from there to Microsoft Excel. The information gathered by that query is explained in the next table.

Table 9 Information gathered about target tables.

Information	**Explanation**
Schema name	Oracle schema
Table name	Table name in schema
Attribute name	Column name in table
Column comments	Column description by developers, if given
Data type	Oracle data type

Information	Explanation
Nullable	Can column be null?
Rows	Count of rows in table
Distinct values	Distinct values in column
Low value	Lowest value in column
High value	Highest value in column
Number of nulls	Number of null values in column
Number of buckets	Number of buckets if histogram statistics
Last analyzed	When the table was analyzed
Sample size	Sample size in analysis
Difference	If > 0, column statistics are sampled, not necessarily representing all rows

Some errors were encountered with the SQLDeveloper export command, because it sometimes converted valid values for "high value" and "low value" to garbage, so it is important to verify all the values in each step to be still valid to avoid problems at later stages.

The final selection of attributes in Excel was done based on domain knowledge by Lean System Development Manager. Attributes selected were very general in a sense that they are all used by most of the customers; only some specialized way of using Lean might add usable attributes to the selection. The selected attributes are presented in the next table with their brief description and they are also divided to categorical or numerical. This typing affects algorithm applicability and data preparation.

Table 10 Selected attributes for DM.

Source	Attribute name	Description
Stock events	COMPANY	Logical company name; categorical
	ITEMID	ID for item in question; categorical
	LOCPRICE	Exact price of item; numerical
	RECID	Unique row identifier (key); numerical

Source	Attribute name	Description
	RECTYPE	Logical deletion status; numerical
	STDPRICE	Standard price; numerical
	UNIT	Unit of item; categorical
Purchase invoice details	COMPANY	Logical company name; categorical
	CURRENCY	Currency the row is in; categorical
	ITEMID	ID for item in question; categorical
	PRICE	Exact price of item; numerical
	PURPRICE_FACTOR	Multiplier if price very small; numerical
	RECID	Unique row identifier (key); numerical
	RECTYPE	Logical deletion status; numerical
	UNIT	Unit of item; categorical

The figure below shows a real histogram for the LOCPRICE attribute with a sample size of 921. This gives some indication about the distribution, but the bin ranges are still quite coarse to be used as a basis for analysis.

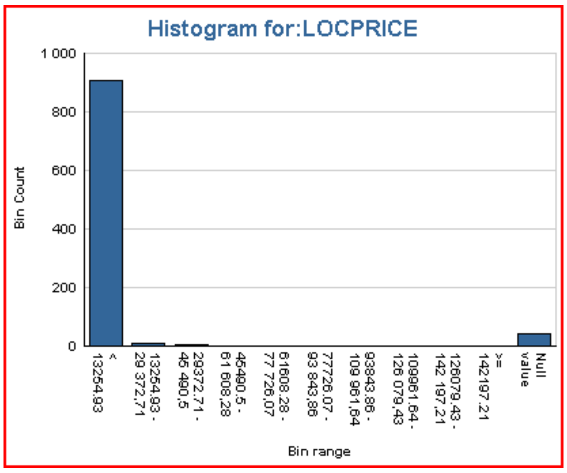


Fig 8 LOCPRICE histogram from ODM GUI with sample size of 921.

The next table present real statistics for one attribute and one database table giving examples of collected information. This gives an idea of the problems related to dimensionality, since there are over 43000 items in use. For some algorithms all those have to be exploded to binary values giving a huge amount of dimensions to tackle, while units and other attributes are much more easily handled. These statistics also caused the STDPRICE attribute to be dropped out of selected attributes, since it had so few values that it was not interesting for DM. Null column signifies if the attribute is possible to be null.

Table 11 Sample attributes and their statistics.

Attribute	**Null?**	**Rows**	**Distinct**	**Low**	**High**	**Nulls**
COMPANY	N	119044	5	n/a	n/a	0
ITEMID	N	119044	43580	n/a	n/a	0
LOCPRICE	Y	119044	16762	-8391.76	5000000	5503
RECID	N	119044	119044	1	119044	0
RECTYPE	N	119044	2	0	1000	0
STDPRICE	Y	119044	1	0	0	74825
UNIT	Y	119044	24	n/a	n/a	0

The target attributes, i.e. targets of outlier checking, were selected to be price and unit attributes in both tables. Deviations from correct values in these attributes had caused problems with monetary calculations on customer sites thus it is of interest to notice outliers in them as early as possible. The author questioned how these values could change since they originate from item tables, but due to flexibility of application they apparently can be changed later and this seems to be very common.

After selecting the source and target attributes, it was necessary to check their general data quality problems against the table provided in 2.7.2 and preparation needs against the table in 2.7.3. This was done with an SQL query mentioned earlier and ODM GUI functionality. The noticed data quality problems were collection errors, outliers, missing values and inconsistencies. Histograms showed skewed distributions concentrated near zero in prices and null values were found in attributes where nulls should not have been possible. Case differences were also found in units which are be interpreted as different categories in DM if not handled manually. Other found problems were negative prices, values 5 000 000 times larger than the average value and many logically deleted rows. This meant that there were already the same kind of errors in the source data that were sought with the DM in the new data. This was an expected result and had to be taken into account in data preparation.

Below is an example of the encountered data quality problems with imaginary values and items. LOCPRICE, STDPRICE and UNIT should always have a value, but they are still sometimes nulls. Once, the unit of item IT313 has changed to "PCS" instead of "KG" and there is also one item having much larger price than others, which makes model building much harder when all others are near zero.

Table 12 Examples of observed data quality errors.

RECID	COMPANY	ITEMID	LOCPRICE	RECTYPE	STD-PRICE	UNIT
1103	MyCorp	IT313	0.1	0	0	KG
1104	MyCorp	IT313	0.1	0	0	
1105	MyCorp	IT313		0		KG
1106	MyCorp	IT313	0.15	0	0	PCS
1107	MyCorp	IT314	285900.00	0	0	TON

Data set exploration and gaining understanding of data was very beneficial because it revealed that Lean System version data set was completely useless for DM testing. It had too few rows and in addition they did not represent realistic usage. The customer data set was not from current Lean version, but its contents were realistic, had enough rows and did not have a trivial distribution. The data quality problems in the latter were also realistic, so they gave information about problems in real-life databases and transformations needed in the data preparation phase.

3.5 DATA PREPARATION

The data sets were already in the local Oracle database, so making them usable for DM modeling meant only creating views based on previous attribute selections. Views facilitate easier usage of the ODM GUI and make sure there are no useless attributes causing modeling problems. The CRISP-DM in fact states that attribute selection should be done at this point through significance and correlation tests, but to the author it was a natural part of previous activity. Reason for doing them at this step through tests became apparent at a later stage, when the selections caused some problems during analysis. Apparently enough descriptive attributes were not selected and this based the predictions and classifications on only one or two input attributes. Nevertheless, the selections were not expanded this time, since the selections were enough for checking the hypothesis about outlier detection.

At this point, a decision was made to use the customer data set for training and initial testing with the ODM GUI algorithm wizard, whenever it just would be possible, but later testing was done with a self-created testing data set. Deleted rows still existing in the database were excluded from the views to avoid training the DM model with already logically deleted source data.

The splitting of source data set into training and testing was handled automatically by ODM. The percentages were selected as 60% of rows for training and 40% for testing since they are Oracle defaults and the row count of training tables were appropriate for this. If too much data is used for training, there is a possibility of overfitting the model which means that also the noise and errors are modeled, as is shown in the next figure. This causes the model to respond poorly to new correct data and it can be avoided by sampling the table instead of using all rows. If splitting is done manually, the rows have to be randomized to avoid using, for example, old existing data for training and newer existing data for testing. Randomization failure can make the model predictions wrong for current usage, which highlights the importance of timeliness in training data.

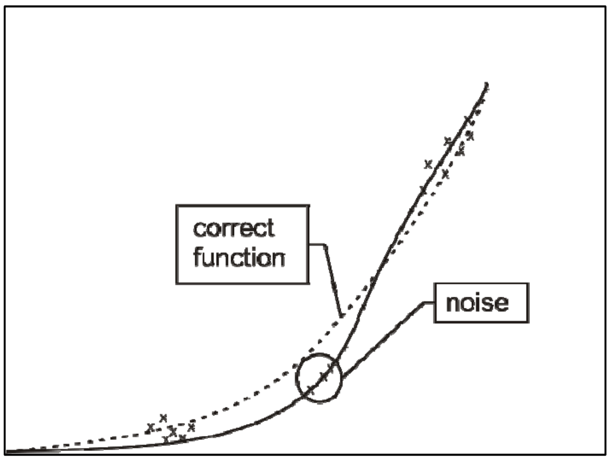


Fig 9 Example of over-fitting, which changes the modeling function to also include noise.

Data cleaning was easier to do on the previously created views because only selected attributes had to be handled. When using whole tables, the SQL queries become much more laborious and error prone to write. Outlier handling, missing values and normalization were needed with almost all algorithms before working with the data. In this case, the actual cleaning was done with the ODM GUI; it had all the necessary transforms built-in. Included transformations ease the preparation phase, since only really complex or rare transformations need manual building with SQL queries.

Outliers were replaced with either edge or null values and they can be identified by their difference compared to standard deviation, some percentage of values or pure min-max values. These operations are commonly known as trimming and winsorizing. Missing numerical values were replaced with the mean value of that attribute and categorical values with the mode of that attribute. Normalization was also a very common operation with minimum and maximum values as limits, while other choices would have been scaling or z-score normalization. The first normalization option was chosen if possible, because based on quick tests, others did not give any advantage in this case. Binning of values was also directly implemented in some algorithms, but sometimes the source data set has to be transformed manually from numerical values to only a few descriptive categorical values like low, medium and high. This is especially helpful when the actual numerical value does not help with analysis but knowing its rough class is very useful.

The order of preparing steps is also important, since extreme values affect calculating the mean and mode in missing values, thus outliers should be handled before that. Normalization, on the other hand, is not possible before there is a value in all attribute rows, so it should be done after missing value replacement. Discretization, binarization and other transformations also have their logical place usually later in the process, but discussing all of them here is not practical because they depend on the source data set.

The handling of outliers in the sample table is shown below with the changed values emphasized and the motivation to handle them comes from their ability to distort binned values and normalization. In this example outliers are replaced with edge values and the cut-off point is standard deviation from the mean in both directions, i.e. values smaller than average minus standard deviation and values larger than average plus standard deviation. Oracle suggests using three times the standard deviation as default, but in this case that caused already present outliers to stick in the training data. Winsorizing would have replaced the outlier LOCPRICE with maximum value of 95th percentile and trimming with null, which in turn would have been taken care in the next step handling missing values. RECTYPE and STDPRICE attributes have been left out as unnecessary and, although this is not statistically a good example, this shows the change caused to source data.

Table 13 Outlier treatment in DM, new values emphasized with bold typeface.

RECID	COMPANY	ITEMID	LOCPRICE	UNIT
1103	MyCorp	IT313	0.1	KG
1104	MyCorp	IT313	0.1	
1105	MyCorp	IT313		KG
1106	MyCorp	IT313	0.15	PCS
1107	MyCorp	IT314	**214425**	TON

Next a table showing missing value transformation is presented. The missing UNIT is replaced with mode and the missing LOCPRICE with mean. The value which evidently should be 0.1 is replaced with a value over 500 000 times larger, which clearly shows that outliers should have been treated more vigorously with this distribution.

Table 14 Missing value handling in DM preparation.

RECID	**COMPANY**	**ITEMID**	**LOCPRICE**	**UNIT**
1103	MyCorp	IT313	0.1	KG
1104	MyCorp	IT313	0.1	**KG**
1105	MyCorp	IT313	**53606.34**	KG
1106	MyCorp	IT313	0.15	PCS
1107	MyCorp	IT314	214425	TON

Normalization helps in reducing the effect of large absolute values in algorithms. This can be achieved with a min-max approach as shown in the next table. It transforms all values to a range from 0 to 1 or some other selected suitable range. The equation used to calculate min-max normalization is shown below.

$$Norm(x) = \frac{x - \min(x)}{\max(x) - \min(x)} * (new \max - new \min) + new\ min$$

Eq. 1 Min-max normalization.

New max and min are 1 and 0 respectively, if the new range is from 0.0 to 1.0. Z-score normalization is presented next and it normalizes most values to a range from -1 to 1, still leaving outliers to be outside that range while min-max hides them.

$$Norm\ (x) = \frac{x - mean(x)}{\sqrt{Var(x)}}$$

Eq. 2 Z-score normalization.

There is also a third approach called scaling, where *x* is divided with the maximum absolute value of attribute, but that is rarely used since it actually does not compress the distribution. The next table shows the effect of min-max normalization for the sample data. As seen, most of the small values compress to a really small range near zero while the largest value becomes 1. Z-score versions are in the additional column which does not cause so much compression but instead leaves the outlying value further away.

Table 15 Effect of min-max normalization for the sample data.

RECID	COMPANY	ITEMID	LOCPRICE	UNIT	ZSCORE
1103	MyCorp	IT313	**0**	KG	**-0.577350454**
1104	MyCorp	IT313	**0**	KG	**-0.577350454**
1105	MyCorp	IT313	**0.25000007**	KG	**2.15404E-08**
1106	MyCorp	IT313	**2.33182E-07**	PCS	**-0.577349916**
1107	MyCorp	IT314	**1**	TON	**1.732050802**

Discretizing could be used on top of previous transformations and sometimes it is even compulsory with some algorithms if there are too many distinct values for an attribute. It makes the model creation easier and faster to build, but makes the accuracy worse. Discretization puts the values to some amount of bins and there are several approaches to binning like equal width, quantile and Top-N. The first two are suitable for numerical attributes and the last one is for categorical attributes. Equal width means that the values are put to bins of equal length while quantile means that there are equal amount of values in each bin. Top-N divides the rows to bins ordered by their value frequency and puts all the rest to a final bin.

The sample data below is equal width binned to 4 bins by LOCPRICE. Each bin width is 0.25 and as can be seen, most of the cases end up in bin number one. With this kind of training data the model predicting outlier possibility is simple since the number of different values is much smaller than without binning. The same kind of binning could be done to units but since there were so few distinct values and they are already discrete, it would not make the model any simpler. It also would not give any new valuable information about the distribution of units, except for the knowledge about rare units.

Table 16 Equal width binned LOCPRICE values.

RECID	**COMPANY**	**ITEMID**	**LOCPRICE (BIN)**	**UNIT**
1103	MyCorp	IT313	**1**	KG
1104	MyCorp	IT313	**1**	KG
1105	MyCorp	IT313	**2**	KG
1106	MyCorp	IT313	**1**	PCS
1107	MyCorp	IT314	**4**	TON

There were also other implicit transformations done by algorithms like correlation transformation by GLM, exploding categorical values to binary and dropping null columns. Some data set distributions might also need training views built on small subsets of actual data set, like price attribute with most of its values between 0-3, while it still had negative values from -8 000 to positive values near 5 000 000. In this kind of case one training view should be for values from 0 to 3 and one for other values to enable building a working model for each value range. Other choices for possible transformations are constructing new attributes from existing ones, merging attributes or, for example, reformatting current attributes with upper case conversion.

Exploding categorical attributes to binary is also known as binarization. It is a very common and important method for algorithms to deal with categorical attributes, since they can often handle only numerical attributes. In practice binarization means that each categorical attribute value is changed to a column with that value as its new attribute name. The only allowed case values are one and zero representing if the original row had this categorical value or not. The sample data is utilized again below as an example of this transformation and it displays clearly the growth of dimensions from one to three for UNIT attribute.

Table 17 Example of binarization of the UNIT attribute.

RECID	COMPANY	ITEMID	LOCPRICE (BIN)	UNIT_KG	UNIT_PCS	UNIT_TON
1103	MyCorp	IT313	1	**1**	**0**	**0**
1104	MyCorp	IT313	1	**1**	**0**	**0**
1105	MyCorp	IT313	2	**1**	**0**	**0**
1106	MyCorp	IT313	1	**0**	**1**	**0**
1107	MyCorp	IT314	4	**0**	**0**	**1**

Some algorithms like SVM support giving presumed outlier rate as a parameter but finding out the outlier rate from source data set might prove very laborious. The difficulty comes from the fact that the dimensionality of attribute space can be huge, so it is simpler to try to remove outliers from the training data by filtering and outlier handling. In simple cases with only a few attributes, i.e. low dimensionality, a histogram might be useful for analyzing an outlier containing attributes one by one, but exploding categorical attributes probably means a huge dimensionality explosion even if there are only a few attributes in the view used. Clustering is one other viable option for checking if the source data contains outliers by checking what the rare clusters contain.

Filtering the source data from redundant and unnecessary information is considered the most important part of the data preparation. It should be done as strictly as possible, because it eases the model building in the next step and might affect the whole outcome of the DM process. This can be achieved by building database views to avoid creating duplicate data in the database via new tables. In this case, logically deleted items were excluded but extra companies and testing material could have been excluded too in order to ease the recognition of outliers in pure production data. Some DM algorithms like clustering and classification are also possible means for identifying data to be removed.

4 MODELING

This chapter discusses the actual modeling process step, explaining the possible choices, activities related to this step and algorithm differences in modeling. In addition, testing possibilities and algorithm performance indicators are explained.

4.1 MODELING PROCESS

Mining models can be created through the ODM GUI if a programmatic approach for it is not required either. For this thesis the ODM GUI was sufficient and in practice the modeling was very easy with it. The ODM GUI has wizards and forms for all necessary input to make sure nothing is forgotten during the modeling. First it asks for data source and column selection. After that input variables, unique key and possible targets are selected in addition to giving a name to the model. As a final step the wizard allows a user to change advanced options for each algorithm and these options regularly have at least sampling, data preparation and algorithm parameter settings. Classification and regression algorithms also have testing options included here.

The GUI also makes it possible to restart modeling from some step by changing only that step's parameters, for instance, after building a model, outlier parameters can be changed and the modeling restarted from that part onward. Complete models are also possible to recreate easily if the source data set changes. The models built quickly even with modest resources like the laptop used in this thesis, where the average build time was about 109 seconds with about 119 000 rows before splitting. In a live production database table there might be millions of rows, but as told before, it is not recommended to train with too much data because of overfitting and the computing resources are more powerful there.

If data sets are very large, sampling of the source might be required to diminish DM model building time, in addition to avoiding overfitting. This requirement depends on the needed performance for model building, but in this case the model is built every now and then, like once a day. Since performance was not an issue and the row count was relatively low, sampling was disabled in all models if not automatically enabled by the algorithm.

After model creation it is possible to create a separate test activity for classification and regression models, if it was not already included in the model building. Its function is to test the model against known target values and compare them with the new predicted values to gain an understanding of the model accuracy.

Regression model tests calculate mean absolute error and root mean squared error (RMSE) in addition to predictive confidence. The last one tries to convey how much better the model is in its predictions compared to pure chance. A residual plot is also drawn and it shows the difference of actual and predicted values graphically. All of these characteristics are useful for evaluating the model performance for input verification; for example, residual plot shows clearly if some ranges of target attribute distribution have significant false predictions.

Classification model tests have slightly different indicators for accuracy. In addition to predictive confidence, they also calculate the actual accuracy by showing the percentage of correctly predicted values for each class. Other descriptors are confusion matrix, Receiver Operating Characteristics (ROC) and lift charts. The confusion matrix shows the number of false and correct predictions compared to correct values, so it expands the accuracy indicator but does not calculate percentages for accuracy or error rates. ROC charts try to give a graphical presentation for the optimum point of probability threshold to use for a classification decision and separating capability. Finally, lift charts try to show how much better the model is than random prediction.

Clustering and anomaly detection methods had to be tested by applying them to the manual test data set defined later. This is also possible through the ODM GUI, thus causing no need for switching to a programmatic approach. The reason for the missing built-in testing functionality for anomaly detection is that it is impossible for the algorithm to know which are really correct and which anomalous values. It does not know if the prediction went right or wrong and is based on the input attributes selected, thus leaving the verification of the result to the user. Clustering, on the other hand, computes rules for each cluster and assigns inputs to its clusters based on these rules. It either does not know if its decision was right or wrong, it just knows that some entries had a small enough distance or density concentration to be placed in the same cluster.

As a consequence, no error indicators are available for either of these methods, except for the support and confidence in clustering and probability in anomaly detection. Support tells the percentage of cases for which the rule holds and confidence is the probability of a case described by a rule to actually be assigned to a cluster. Clusters can also change between different runs of modeling, since they have a random starting point selected in the beginning, so these values are not constant with the same training data set. Anomaly detection probability gives an estimate of how good the outlier prediction was according to the algorithm. These cause the evaluation of classifying trustworthiness and clustering results to be left to the user.

When the model is complete, an apply activity can be built. This activity is also known as scoring a data set, where data set is from one value to a complete table or view. Data scoring means feeding new data to an algorithm so that it can predict the results for some attribute or describe the new data based on the model created. First the fed data has to be prepared exactly the same way as the training data. This means that the input data has to go through the same normalization, outlier, missing value handling and other possible transformations as the training data did, otherwise the predictions and descriptions are wrong. If the ODM GUI is used for scoring, it will use the knowledge from the model to apply all necessary transformations before applying the actual model. If the anomaly detection method is used, outlier handling can be left out, but with others it is still necessary. The results of this activity are the predictions and descriptions of the fed data, depending on the algorithm. A probability is also being published for some of the predictions and descriptions for the user to make an educated decision about trusting them.

4.2 MODELING ALGORITHMS

Modeling required checking of all the Oracle algorithms and methods for applicability to input checking problems. At this point of the process, the suitable modeling techniques seemed to be classification, regression, attribute importance, clustering, feature extraction and anomaly detection. Feature extraction was only applicable if some attribute was not suitable for using as is and it would have been possible to create a new attribute from it. The attribute importance technique was only suitable for attribute selection if no domain expertise would have been available. It shows how much other selected attributes have information about the selected attribute, but its results are not necessarily usable for attribute selection as such.

Available algorithms for these modeling techniques were Naïve Bayes, GLM, SVM, K-Means, O-cluster and MDL, which were listed earlier in 2.6. Assumptions for modeling were that all data preparation could be done using automatic embedded data preparation in ODM. Later this proved to be false for some distributions in the price fields and manual data preparation was necessary with the help of the ODM GUI. A programmatic approach to preparations was not necessary.

Before explaining the algorithm differences, it is necessary to explain how the algorithm results are used for outlier detection. The detection is done when a user inputs new value, but the comparison against previous values depends on the chosen method. From the available ones regression and clustering can be used to check continuous values while classification and clustering suit categorical attributes.

Regression algorithms can predict what would have been the correct value with some error tolerance and the implementation could cause an alarm if the difference is large enough. Classifying algorithms can give a possible class to an entry based on the source attribute values, so enough other attributes have to be available to predict the correct class. This predicted class is then compared with the user input and user is alerted if they are different and the probability for the prediction is good enough. As an exception, anomaly detection with SVM classifies the new entry directly to either zero-class representing outliers or one-class representing normal values and gives a probability for this classification too. And finally clustering algorithms place the new entry in some cluster giving also membership probability, based on the source attribute values.

The only problem in the last one is that the cluster number gives no indication of user input correctness, since the previous cluster number for a given source attribute values is not known. The probability as such tells that probably similar entries have ended in this cluster before, but that is not certain. This comparison might be easiest to do by saving the previous cluster number to some other database object, for example an extra table, attribute or predictive function based index. This assumption is based on the fact that running DM model on all present cases in the database would require too much time to be usable on the fly.

Almost all tested algorithms were suitable for input validation, but there were interesting differences between them. As said, it was possible to apply the regression method to the prices, them being continuous numerical values. Both GLM and SVM algorithms worked, but both of them also proved to be very arduous to train with the available distributions. Regression algorithms give numerical predictions for some dependent attribute based on other independent attributes given as input. With simple linear distribution, GLM gave good predictions which were within error boundaries. When the model was trained with complete distribution, it was nowhere near the correct values and the same results applied to SVM in regression.

None of the available parameters helped with this behavior, and the solutions arrived at were using several piecewise linear models, reducing the amount of independent attribute values and adding more cases for each attribute value. The first one helped to get the predictions right but the second and third solutions were not tested. Other inaccuracy causing properties of the distribution were some items having several prices and not having enough descriptive independent attributes. The next table has an artificial sample prediction with probabilities to show how the prices are closer to the correct ones near the dominating ranges and further away from the correct price at the outer regions. The table also demonstrates how the probabilities are lower when the algorithm is less certain its prediction is right.

Table 18 Price predictions with the probabilities for the predictions being right, calculated from the model.

RECID	**COMPANY**	**ITEMID**	**LOCPRICE (predicted)**	**PROBABILITY**
1103	MyCorp	IT313	0.120	0.85
1104	MyCorp	IT313	0.120	0.85
1105	MyCorp	IT313	0.120	0.85
1106	MyCorp	IT313	0.120	0.85
1107	MyCorp	IT314	350000.00	0.34

Classification tries to fit some class label to the given attributes and Oracle provides Naïve Bayes, GLM and SVM algorithms for this purpose. Classification suits mostly binary or nominal categories and not numerical or ordinal attributes (Tan et al., 2006) and, as a consequence, it is not able to create classes for prices. As a surprise almost all algorithms classify cases to units quite well, even with only one high dimensional source attribute. At this point, it must be clarified that it was not possible to use GLM classification to predict anything since it requires only binary categorical attributes and there was no possibility to binarize the source data sets due to time restrictions. The performance of Naïve Bayes and SVM was average, except with less frequent values, but the main problem was that there was not enough descriptive attributes available to really predict the UNIT accurately in most cases.

At first, it was also believed that the Decision Tree algorithm is not suitable for outlier detection and it was, therefore, excluded from all previous plans, but in fact it is a classification which provides predictions and explains them with rules. As such it could have been extremely usable for the purpose of this thesis, but in the end it was not, since the rules were not able to base themselves on ITEMID and other descriptive attributes were not available, as stated earlier.

Anomaly detection is also a classification method, but it is unique in classifying only to the classes zero and one, which is why it is also known as one-class-classification. Oracle uses a very advanced SVM algorithm for this and with some distributions it apparently works. However, in this case it did not provide much help in finding outliers, since dimensionality grew very large in categorical attributes - another possibility is that with enough training data it would model even the current distribution. With numerical attributes it worked when the difference between correct and incorrect value was large enough, i.e. 10 000 times the right value. This caused it to be practically useless in predicting except for those numerical values off by several orders of magnitude, since probabilities were often very low and a guess would have been as good as its prediction. Another possible accuracy enhancing procedure could be the division of training data set to only a few items at once to simplify the classifying, but this would increase the amount of models.

A sample table is again provided showing new input for LOCPRICE and a prediction for the class of one or zero, where the first is normal and the second an outlier, respectively. The probabilities reflect the algorithm's view of prediction correctness. As seen, the last prediction is quite far from the correct value (285 900), but the algorithm seems to be unsure about its decision due to the value's absolute order of magnitude.

Table 19 Sample table of the one-class classification for anomaly detection.

RECID	COMPANY	ITEMID	LOCPRICE (new input)	PREDIC-TION	PROBABILITY
1103	MyCorp	IT313	**0.120**	**1**	**0.85**
1104	MyCorp	IT313	**2.000**	**1**	**0.85**
1105	MyCorp	IT313	**120.000**	**0**	**0.85**
1106	MyCorp	IT313	**-100.000**	**0**	**0.85**
1107	MyCorp	IT314	**214420.000**	**0**	**0.34**

Clustering with K-Means and O-cluster algorithms was quite like classification. They do not have targets but build clusters based on distance calculation or density. As a preprocessing step, or in data exploring, clustering can be easily utilized for outlier detection by searching for clusters with the smallest amount of members, i.e. outliers. This can provide valuable insight into the data being mined. In outlier detection through comparison to earlier values, classification seemed to outweigh clustering's vaguer results, thus there seemed to be no real reason to use clustering for this purpose. This was proven wrong when it was apparent that clustering works for predicting outliers for continuous attributes and the results are in fact quite well-defined. This is discussed more thoroughly in Chapters 5 and 6, which cover measurements and evaluation respectively. The following table shows an example of clustering with some simple rules and clusters. Classification would be somewhat similar except for the clusters being classes with known values.

Table 20 New inputs clustered according to the rules formed by training.

R.ID	COMP.	ITEMID	LOCPRICE (new input)	CLUSTER	RULE
1103	MyCorp	IT313	**0.120**	**1**	**LOCPRICE >= 0 AND <=2**
1104	MyCorp	IT313	**2.000**	**2**	**LOCPRICE >= 2 AND <= 20**
1105	MyCorp	IT313	**120.000**	**6**	**LOCPRICE >=85 AND <= 150**
1106	MyCorp	IT313	**-100.000**	**17**	**LOCPRICE <= 0 AND >= -2500**
1107	MyCorp	IT314	**214420.000**	**14**	**LOCPRICE >= 200000 AND <= 300000**

Preprocessing selections were mostly Oracle defaults, so, for example, outliers were first defined as values three times the standard deviation away from the average value and replaced with nulls or edge values. Missing values were always replaced with the mean value or mode for a value, depending on the attribute type and normalization was always min-max, since none of the distributions was enough Gaussian-like. Only changes to preparation choices were made to binning selections, since sometimes algorithms failed if binning was done and sometimes they failed if it was not done to some attribute.

4.3 PROGRAMMATIC APPROACH

In programmatic approach, all DM activities can be done through PL/SQL packages like DBMS_DATA_MINING and DBMS_DATA_MINING_TRANSFORM. There is also a package called DBMS_PREDICTIVE_ANALYTICS for ad-hoc searching of significant attributes capable of explaining the target attribute and predicting one. Functionality available through these packages was sometimes poorly documented and that caused delays in using them, especially with custom transformations. It was also noticed that packages required a dot as a decimal separator or they would crash the session, which I presume is a bug in Oracle. In practice this meant changing regional settings to match correct separator, which is not a common way of working in most applications.

Information about created models is available through static Oracle dictionary views like ALL_MINING_MODELS and ALL_MINING_MODEL_SETTINGS, but some information has to be retrieved through the previously mentioned packages. Sometimes the ODM programming environment appeared to be incomplete because of completely different access methods for the same kind of information. As an example, algorithm parameters were available from static view but transformations had to be retrieved with a package. Perhaps maturing of this technology in databases will cause simpler views to be built for accessing this kind of data.

All models are easily usable in SQL with functions for prediction, clustering and feature extraction. For example, the PREDICTION function returns the best prediction for the target attribute based on given input attributes and PREDICTION_PROBABILITY gives the probability for that prediction. Packages can also be generated for models originally created in the ODM GUI. The packages contain the embedded transforms which simplifies the process and this is probably the best method to deploy the functionality in the actual application.

5 MEASUREMENTS

This chapter explains the measurement plan evolution, target setting and publishes the testing results.

5.1 MEASUREMENT PLAN

The measurement plan was to take a production data set and test all algorithms against that with ODM's automatic testing function. The first approach of using production data for testing arose from the wish to find outliers of the current production data. During modeling this changed to manually created test data set, since otherwise it would have been impossible to accurately define what the test cases were to measure the actual accuracy. One other reason for discarding this test method was that it failed to yield a clear picture why something ended up as an outlier in the customer data. This would have required too much interaction with the data owner to be possible during this thesis.

The first manual test data set was created by first choosing random ITEMIDs from the training data set and creating cases containing correct and incorrect values for each tested attribute. Incorrect values were attempted both as solitary errors and combinatorial errors, since the models were built by including most of the selected attributes as inputs. After noticing that combinatorial outliers were not possible to find with one model, the first manual approach later diminished to a second manual test data set with variations in UNIT attribute and separate cases for price variations. The selected items were also changed to most frequent ones for each unit, because random units and prices did not surface well enough in the algorithms.

The final third compilation of tests did not contain any errors in UNIT attributes but instead asked the algorithms to classify the cases based on other attributes and the result was compared with training classification. The price cases had conscious order of magnitude errors, but did not actually need them before clustering appeared as a viable algorithm in addition to prediction. The measured value was the correct predictions, which was later divided by the amount of test cases and reported as a percentage of correct answers. The correct values were included in the test data to get a comparison target and subtracted from the amount of test cases before division.

In the third manual test data set, the numerical attribute, i.e. LOCPRICE, had correct values and order of magnitude errors ranging from 0.1 to 1000 times the correct value. Categorical attribute validation did not in fact require any value for UNIT, except for anomaly detection, since it changed to problem of classifying or clustering ITEMIDs to correct classes or clusters before it was known what is proposed as the case's new value. Anomaly detection tried to predict directly if the wrong UNIT or LOCPRICE was given, and the result was calculated for UNITs by counting the correct UNITs classified as anomalies. Experimenting with earlier test data sets already gave a good picture of the anomaly detection's abilities, because it always weighted dominating coefficients more, no matter if they were UNITs or ITEMs and it was already clear that it could not perform with the used source data without some unknown transform. This also caused prices to be irrelevant for classifying.

5.2 TARGET SETTING AND TEST RESULTS

A target was set of getting 80% of price predictions correct with regression for five most frequent items, when the error bounds are taken into account and multiple prices are allowed per item. With classification, the accuracy target was set to 100% with units occupying ~90% of the cases, which in practice means only four dominating units. Classification had to do this to over 40 000 distinct items based on their ITEMID, so it was also not expected that all would be correct. The dominating units were tested with five items selected uniformly from each one's distribution with the lower bound for each being ten items. If an item had less than ten cases, it was discarded. Finally, clustering used both of the previous targets, because it was usable for both continuous and categorical attributes.

The best achieved results are published in the following table together with an evaluation of whether the algorithm is usable for anomaly and error detection with some future modifications. Some comments are also presented to clarify the results. "Yes" means that the previous targets were reached and, therefore, "no" means the opposite. Parentheses signify a possibility to enhance the result with some transform and "n/a" stands for not applicable for that purpose.

The percentages in testing are quite coarse since only 17 test cases were created for units and 25 for price, from which all were for clustering and five for regression. The price cases had also correct answers included to get a reference value for clustering and regression. The low amount of test cases is due to the fact that these cases covered most of the dominating units and items in the database. The rare values were not tested since it is very unlikely that they would be modeled with constant performance, because algorithms concentrate on most frequent values.

Table 21 Measurement results.

Method / Algorithm	**UNIT results** **(LOC)PRICE results**	**Comments**
Classification / Decision Tree	No, 29% correct predictions n/a	Predicts some right, but bases prediction on price which does not significantly predict unit. Useless for predicting in this case since ignores ITEMID. Probably too few descriptive source attributes.
Classification / Naïve Bayes	Yes, 100% correct n/a	Model shows surprisingly in automatic modeling tests that one of dominating four units is predicted right with only 40% of the cases, even though manual tests show better results. Probably too few descriptive input attributes and too many dimensions.

Method / Algorithm	**UNIT results** **(LOC)PRICE results**	**Comments**
Classification / SVM	Yes, 100% correct n/a	Very dependent on algorithm parameters, “max. overall accuracy” much better results than “max. average accuracy”; probabilities of predictions low. Too few descriptive input attributes and too many dimensions.
Classification / GLM	n/a n/a	Classifies only binary categorical attributes, not suitable for this case.
Classification / Anomaly detection / SVM	(No), 76% correct. Noticeably errors based on rarity of item or unit. (No), 0% correct. Because the units automatically weight more.	The algorithm gives most weight to well-classifying attributes like UNIT. This causes the classification to be based almost solely on UNIT value and rare values are also outliers even when they should not be. As a side-effect price outliers were not noticed at all. Leaving UNIT out causes too many wrong predictions to be usable either.
Clustering / K-Means	Yes, 100% in separate clusters, when more clusters given to algorithm than there are units. (No), 60% correct.	Finds clusters based on UNITs as classification and tells which the rare ones are. As such suitable for analyzing data quality, but classification does this better. Price clustering extremely usable for finding outliers thru histogram-like concentrations. Problem that creates given amount of clusters even when no real clusters are present and finds differences beginning from 100 times larger values. Could probably be enhanced by changing parameters, but O-clustering’s directly better performance makes this unnecessary.

Method / Algorithm	**UNIT results** **(LOC)PRICE results**	**Comments**
Clustering / O-cluster	(No), 54% in separate clusters, even though number of clusters given to algorithm same as for K-Means. Yes, 100% correct.	Finds clusters as K-Means clustering does, but puts some UNITs to same clusters turning results unusable. Price clustering much better since creates only needed amount of clusters and seems to have better separating capability in dense areas. One noted problem in literature (Milenova & Campos, 2002) with creating artificial clusters for low dimensionality data, which was not a problem with the test data set.
Regression / GLM	n/a Yes, 100% correct predictions within Root Mean Square Error of correct answer.	Predicts price when training distribution is linear enough. Enhancements might need classifying items to some price ranges and training with them, building separate models for each linear distribution or adding more independent predictor attributes. Compared prices were averages from actual values.
Regression / SVM	n/a Yes, 100% correct predictions within Root Mean Square Error of correct answer.	Predicts price when training distribution is linear enough. Enhancements might need classifying items to some price ranges and training with them, building separate models for each linear distribution or adding more independent predictor attributes. Compared prices were averages from actual values.

These measurements show the classification, clustering and regression methods to be usable for outlier detection and input validation in ERP systems. From the successful algorithms, the Naïve Bayes and SVM classifications showed excellent accuracy in manual testing and thus can be recommended for categorical attribute analysis, but only if more descriptive source attributes are found. This is due to the fact that the automatic testing during modeling showed surprising errors with even dominating units. Source attribute dimension reduction could also improve algorithm accuracy. Decision Tree and GLM classifications were, on the other hand, completely unusable since Decision Tree based its predictions only on price due to the same lack of descriptive source attributes and GLM needed binary categorical attributes.

SVM and GLM regressions showed good performance in predicting prices but their prerequisite was that the distribution was divided into ranges where each member resembles the others and this is easily achieved with clustering too. If division was not done, the RMSE values grew to thousands instead of being near two, which makes the model worthless when correct prices are mostly between zero and one. It is the expected result that the outliers inside the chosen range affect calculation of this error measure and trying to fit the whole distribution under one regression model is impractical if there are separate clusters of prices with a huge difference in order of magnitude.

The previous result, on the other hand, shows that clustering can be used to find price groups, or in a more general sense, continuous value clusters for regression. Clustering is also very useful for directly predicting if a continuous value is outside the value concentration, i.e. an outlier, in addition to using it as a way to classify prices to *n* clusters and then labeling them as price groups. After this the comparison of the new value could be done against the group value, stored in a table with the ITEMID. Clustering was also partly successful with categorical attributes, but the difficulty rises from the unknown amount of needed clusters.

When it comes to clustering algorithms, K-Means was very effective in clustering categorical attributes but did not reach the target with continuous ones. A different distance measurement or other parameter change might improve the accuracy to reach the target, but due to the sheer amount of combinations they were not attempted. Another problem with K-Means is that it always creates as many clusters as are parameterized and that causes cluster splitting even when it is not required. The same kind of unnecessary splitting problem applies to the O-cluster algorithm with low dimensionality data, but it still manages to build a more realistic amount of clusters reflecting the actual distribution better. This can be fine-tuned with its sensitivity parameter so that lowering the value of sensitivity causes fewer clusters and apparently it performs better than K-means with high dimensionality data (Milenova & Campos, 2002). As a result continuous clusters were better with O-clustering, but it failed to reach the target with categorical attributes.

The surprising fact was that the one-class classification with the SVM algorithm, fitted for anomaly detection, failed to provide such insight into outliers as would have been expected. Its results were at their best mediocre and only for UNITs. In separate single tests it noticed price differences after the price was changed, for example, to a million times larger, but this is understandable if the price distribution was not divided into smaller ranges as with regression, since it is the same algorithm in question. It would be interesting to see if it performed better with a so called kernel trick (Vapnik, 2000), where the kernel in the algorithm is changed to some kernel suitable for the current distribution, but this would require coding the complete algorithm manually. And it is still suspected that there is no such kernel which could model the current distribution without any division.

Almost all methods and algorithms were dependent on the source data properties. This makes it impossible to give tenable recommendations about preprocessing choices or algorithm parameter settings and therefore they are omitted. They are also not crucial regarding the scope of this thesis.

The results showed that continuous attribute prediction with regression suffered if there were no approximately linear or Gaussian distributions with enough predictive independent attributes. Categorical attributes needed to have a limited number of values to make it possible to use them in classification as predictors and the lack of descriptive attributes implied problems with the attribute selection or collected source attributes in the ERP system. Clustering of categorical attributes was dependent on descriptive attributes too, but clustering of continuous attributes was the only algorithm seemingly performing well with all available source data sets. This is not to say that it would not perform otherwise depending on the source data set in general.

6 EVALUATION

This chapter goes through the problems encountered, choices and interesting observations made during the thesis.

6.1 TESTING

The testing phase is evaluated first, since due to the cyclical nature of the DM process it affects the previous phases, as can be seen later.

Expected errors in input data are wrong values compared to old ones, i.e. wrong prices and units. It is, of course, possible that an item is sometimes sold with a different price and orders might contain either an item or a box of items, but the attempt was to detect deviation from the mode or mean of an attribute for an item. With prices the mean in itself is not enough but some tolerance has to be allowed for the accepted values, for example, standard deviation.

The test design of modeling consisted of splitting training data into training and testing data sets automatically and manually. Additional manual testing data had to be generated to carry out actual measurements with known test cases. This was done by creating a separate table for testing data and inserting suitable rows into it with SQL. The first test cases consisted of finding out the correct category for an item and order of magnitude changes in numerical attributes and combinations of these. During tests it was learned that combinations were not possible or practical to search for and gradually unit and price tests were separated.

At first, there was a timeliness problem with split testing data set due to faulty manual splitting and that was noticeable because the test data set results were worse than with the training data set. As an example, the training data set showed an outlier percentage of 20% and test data set 40%, whereas with valid splitting, the percentages would have been approximately the same. This also highlights the importance of selecting the training data set to be from a period whose usage pattern is similar to the current use. To maintain consistency, the models have to be retrained at regular intervals. In implementation this can be understood as using a sliding training window consisting of a certain amount of time backwards from the current moment. The timeliness issue was fixed by using automatic splitting in ODM GUI.

The second problem arose from the modeling where Oracle defaults were used for outlier handling. This meant that values normally three times the standard deviation were considered as outliers and that left too many outliers in the training data, biasing the models on that part. This parameter was changed to allow only one or two standard deviations before considering a value as an outlier, which should have improved the results with the tested top five items, but apparently dimensionality still dominates the performance of the algorithms affected by outlier removal. This verified the suspicions that it is not possible to make one model predict every possible item without sacrificing the accuracy.

The third problem came up with the manual test data set, since the ITEMIDs selected for testing were accidentally rare ones. Because the algorithms favor dominant attributes and values, the observed predictions of price were nowhere near correct. This was fixed by selecting the test entries to be from the five most frequent items in the table. The numerical values are also rarely exactly correct in predictions since they are continuous in nature, they have several right values for one item and the source data set has large variance. That is not a problem in practice, since ODM gives error estimates and probabilities that can be used to decide if a user has to be alerted. Categorical tests suffered from the same rarity problem and were improved by selecting items having most frequent units uniformly from each unit's distribution.

It should also be noted that statistical methods and hard-coded rules might prove to be better for checking the correctness of values, when not enough descriptive input attributes are available. Statistical methods also work with rare entries, if preliminary values exist in the database. They just have to be limited to a small time frame to avoid scanning the whole database or an additional attribute, table or function based index has to be built for quick retrieval of the correct answer. This approach has its limitations due to dimensionality, as discussed later in Section 6.5.

6.2 DATA UNDERSTANDING, ATTRIBUTE SELECTION AND DATA PREPARATION

It became quickly obvious from the results of testing that only attributes necessary to predict or classify the current target should be included in each model even though there was a preliminary attribute selection of non-redundant attributes. As an example, if some model was to be used in clustering UNIT-attribute, it did not have much use for price attributes, since they contain little or no information about the unit. Extra attributes just complicated the models, making them often more inaccurate than with less attributes, although there is a danger of eliminating subtle predictors for a target attribute. As a rule of thumb, it is, perhaps, best that the selection should be expanded only if there are descriptive attributes helping in identifying the correct class or simplifying the prediction. A good external example of this is the case of a bank loan risk assessment, where income, house size and numerous other attributes can be used to predict if a customer is likely to default.

The other side of the attribute selection problem was the lack of descriptive source attributes, mentioned several times earlier. Even though there should not be any extra attributes, there should be more attributes having information about the prediction target. This is not necessarily only a selection problem, since with ERP there might not be such descriptive attributes at all for all target attributes. As an example, address and country might predict the postal code, but postal code and country will not predict the address with adequate accuracy.

Dimensionality curse was the main factor causing models to fail or to be inaccurate. All methods and algorithms suffered from its effects due to over 40000 distinct items being present. A suitable preprocessing method for grouping them or changing the way of using ERP could reduce the negative effect these dimensions have. In a way this was almost a worst case scenario regarding the amount of items, so better performance is expected with other customer distributions.

The only actual preprocessing problem was caused by outlier handling in ODM. The default threshold value for outliers was so large that too many outliers were always left to the training data set and that caused model error tolerances to become too large. Lowering the threshold improved the error tolerances but it did not solve all accuracy problems, because dimensionality was still such a dominating factor.

6.3 MODELING

Previously, it was also mentioned that the LOCPRICE- and PRICE-attributes had a distribution where most of the values were concentrated inside a small range. These kinds of distributions required separate models for each concentration of values, either through training views showing only the wanted data or through algorithm parameters, where the correct range was identified. Another problem was that some items have several prices and, therefore, regression cannot predict all of them, even when the error tolerances are taken into account. Although it is possible to build a model to predict some prices relatively well with regression, a better prediction would need more attributes which the price is dependent on. It was finally realized that an easier way to model prices is to use clustering with the O-cluster algorithm, since that tells if the price falls into the same density concentration as before. The downside to this approach is that the expected cluster for each item has to be saved somewhere in advance; an additional table, attribute or function based indexes are again a viable suggestion for this.

Nevertheless, it seems to be possible to also build hard-coded rules with statistical functions to mimic data mining functionality in simple cases. Categorical attribute modes and numerical attribute mean values could be stored in additional attributes or function based indexes together with the value of the target attribute. This would speed up the retrieval and remove some load from the database. And even if data mining was used, the author believes that at least clustering would need similar storage for its analysis of past data, since it is not practical to run the mining algorithms on all training data during input phase of a new data. This, of course, depends on the implementation method, but other suggestions were not invented in quick sketches for deployment. In this particular case the UNIT attributes could be checked with static rules against UNIT in the item definition table. This would be much faster and more reliable than any model or statistical calculation, but this was outside the scope of this thesis.

The author still considers data mining better than statistical functions, because it scales better when the dimensionality of input attributes grows from one or two and the number of rows is more than several hundreds of thousands. The advantage of DM is that its predictions are more subtle and can predict correct class or value even when statistical functions are completely wrong. As specified even in the definition of DM, it works where statistical methods are unusable and even though there is certain work in maintaining DM models for each attribute, the indexes are not maintenance free either and become burdensome to maintain when there are too many of them. They also slow down the database, since they have to be updated during the insert and update operations to the tables, which is not the case with DM.

Moreover, the model failures are believed to be the result of preprocessing failures on the author's part. With ideal distributions, all models would have performed ideally, but sometimes finding the right preprocessing transformation to achieve close to ideal distribution was too time-consuming for the project schedule. The importance of histogram and other distribution statistics were also underestimated, even though some of them were collected. By concentrating on the histograms, it would have, perhaps, been possible to notice the difficult distributions of prices beforehand, not at the middle of testing the algorithms. Dimensionality was also noticeable in the preliminary data exploration, but the author failed to understand the ramifications of it before attempting to use the algorithms. The attribute selections could also have been better, since the absence of descriptive attributes should have caused selecting more source attributes or to find new targets. The benefit of making these errors was to actually see, what happens when the wrong choices are made.

Better algorithm understanding would have helped too, but the mathematical backgrounds were so extensive that there was not sufficient time to learn them all with the relatively short period available for studying them. This affected the testing and made the whole measurement process a somewhat trial-and-error –kind of practice. On the other hand, much knowledge was gained about DM algorithms, methods and processes which makes it possible to make more educated decisions in future DM projects. This also retains the expectation of being able to clearly define realistic DM objectives to be more certain of a successful project. Indicators of model quality should be understood well too, since without them the evaluation of models is impossible. This might seem obvious, but the indicators are not always that easy to understand, as is with the case of ROC and lift charts.

Knowing the tools is another important factor in successful DM process. Learning to know a new tool and algorithms might take a considerable amount of time in the beginning and this has to be taken into account when planning project schedules and costs. As a personal experience, the author had a deep distrust of Oracle's graphical user interface and first attempted to pursue a programmatic approach, while in the end the GUI proved to be much easier and very usable for its purpose.

6.4 RESULTS

The test results in ODM for final manually generated test data set are listed in the following table. "Yes" in a cell signifies that the algorithm can be used to identify this kind of outliers in a Lean System ERP database and "no" the opposite. Not applicable (n/a) means that the algorithm could not predict that type of target or required preprocessing transformation which was not available. Parentheses with "no" signify that the author was not able to train the algorithm to notice outliers very well, but different preprocessing or training might still help. Unfortunately, it was not possible to try external transforms due to the limited time frame and even selecting good candidates would have been a completely new study. In fact, there are numerous books and studies about preprocessing methods.

Table 22 Test results for manually generated test data set.

Test / Algorithm Target	**Classification / Decision Tree**	**Classification / Naïve Bayes**	**Classification / SVM**	**Classification / GLM**	**Classification / Anomaly**	**Clustering / K-Means**	**Clustering / O-cluster**	**Regression/ GLM**	**Regression / SVM**
Unit change	No	Yes	Yes	n/a	(No)	Yes	(No)	n/a	n/a
(Loc)price change	n/a	n/a	n/a	n/a	(No)	(No)	Yes	Yes	Yes

Ranking the previous algorithms for each method by their successfulness and ease of usage is quite simple. Naïve Bayes is the easiest to use for classification, since it is based on pair probabilities which are quite simple to understand and are published with the model. When the classification is not successful with Bayes, it is recommended to turn to SVM because of its capabilities with complex and high-dimensional data. In clustering both available algorithms provide understandable division rules for formed clusters, but O-clustering is better than K-Means with high dimensional data and avoids unnecessary splitting. In regression, GLM has probably a simpler mathematical background than SVM, but both are, in the end, based on the regression principle. SVM's advantage over GLM starts showing when modeling distributions with high dimensionality.

The plausibility of these results is very subjective, since all results depend very much on the training data set, attribute selections and preprocessing choices. This dependence is, nevertheless, not important since the basic idea has been verified as working and now it is possible to start constructing attribute selections, transforms and algorithms optimized for this purpose. On the other hand, the reliability of these results is objective from the author's point of view, since the used algorithms were designed for classification, clustering and regression and it is believed that they would not have been included in commercial products if they would not fulfill their purpose in that sense. Nevertheless, it is correct to say that ERP systems can use DM for anomaly and error detection in input validation. The limiting factors for a successful project are the source data set, attributes and preprocessing possibilities.

6.5 IMPLEMENTATION

The results are not directly deployable, but some implementation ideas were gained during the thesis work. These ideas rely on database triggers, or application logic, as a starting point for validation event. Validation of a new target attribute value can be performed when the data has been fed to all or most of the source attributes predicting that target attribute. This implies that the validation can actually be done after all the source attributes are available, but checking the value fed to source attributes might be impossible. This happens if the current target should be used as one of the sources for its own predictor in some other model, so the selection of predicted attributes has to be limited to attributes seriously needing checking. The validation can be done against model answers or values stored in other database objects and the users should be able to decide how the warning is handled. The actual models by ODM are very easy to integrate to SQL clauses, since ODM can create easily usable PL/SQL packages for applying transformations and models. There are also several SQL functions available for integrating DM model dependent functionality.

The means and modes for the previous values could also be calculated with statistical functions for all items every now and then, but that could congest the database because of generated full table scans. As suggested previously, other possibilities could be additional database objects storing the mode and mean for every item's unit and price. Those are not without cost either, because they might also generate too much activity every time they are updated. This is again due to dimensionality and, in addition, the indexes need to be updated every time the indexing target is updated.

Monitoring of accuracy and maintenance are also viewpoints that have to be taken into account when developing the implementation. The accuracy of predictions, classification or clustering has to be monitored somehow and when the model degrades enough, automatic retraining has to occur. The other possibility is to retrain periodically just to be sure that the model is current and the recommended retraining periods depend on the volume of new and changed data. This is because some tables and attributes change several times a day while others stay the same year after year. Training data set selection can be a sliding window representing the last fed data from some suitable timeframe in the history, ending at the present moment. Maintenance consists of at least refreshing and balancing possible indexes or external tables to keep their access speed fast enough. A model becoming obsolete has to be noticed too to avoid unnecessary load for the database.

If the deployment is implemented, the benefits gained from anomaly detection functionality should be measured as a difference of costs incurred by errors in ERP data before and after the implementation. This requires that the current costs caused by errors are somehow known, but that is beyond the scope of this thesis. License fees have to be taken into account in calculating the benefits and they might reduce or eliminate the payback of the investment in some cases to a level where the functionality is not yet profitable to implement. If ODM pricing is too high, Open Source algorithms can be tuned to fit application requirements, but apparently the quality of algorithm implementations is very varied and preprocessing algorithms have to be implemented as well. Additional observations about statistical rules give the impression that they are easier and cheaper to implement and the author is quite certain that they will be looked into in Lean System ERP.

7 CONCLUSIONS

In the beginning, business and DM objectives were defined as a part of the DM process to be equivalent to the research question. These goals were to study if it is possible to verify inputs and find outliers with DM and to find suitable algorithms for this work. These objectives were reached and they are likely to produce cost savings for companies in the future through data quality improvement, as was stated in (Eckerson, 2002). All the sub-steps were researched to make sure the result is extensive and repeatable.

Other requirements were related to comprehensibility, deployment and price tag attached to developing this kind of functionality. The assessment is that the results were simple to understand and some deployment ideas were gained. Common data input problems were also detected, as was expected, and statistical rules were noticed as a successful simple method for additional sanity checks in an ERP system. The reasonable price was the only target impossible to reach with Oracle Enterprise Edition with the ODM option. It is also not known whether Open Source alternatives could be cheaper than Oracle due to development costs, and whether their performance is at the same level.

Some criteria for the algorithms had to be relaxed in the modeling phase. For example the dimensionality curse became suddenly a problem that could not be easily avoided, since the algorithms created the problem by exploding tens of thousands of attribute values to new binary attributes. Training rarely changed to training frequently, since the actual training was not that burdensome for the database and changed usage patterns had to be modeled regularly. Robustness against over-fitting is actually not known, since testing that algorithm property was not within the scope of this thesis. Algorithms seemed to favor dominating values which gives rise to the thought that they do not fit errors to the model easily, if the errors are not dominating the data set. As a downside this also makes it impossible to use the models for rare values, so it is presumed that a complete coverage for all values is also impossible. This is, however, not a problem since the idea was to check inputs against common present values. The criterion for scalability was fulfilled to the extent of available test material and the built-in functionality criterion was completely fulfilled.

It is possible to notice and warn about outliers in input data, but transforming the data sets to a suitable form for a successful DM project might prove to be laborious. Testing, data understanding, preparation, attribute selection and modeling can all have unexpected problems if understanding of some component is not at the required level. Personally, during this thesis work, the author encountered at least a poor testing plan, dimensionality curse, wrong preprocessing parameters and poor attribute selections in addition to too light understanding of mathematics behind the algorithms. On the other hand, now the knowledge about these factors is imprinted in the mind of the author.

Attribute selection and preparation of data sets are the most important and time-consuming phases, as was said in (Pyle, 1999). Without acquainting oneself with the source data, it is quite impossible to estimate the time required for these phases and gives rise to the possibility of exceeding project estimates. Algorithms could be optimized for this purpose by building easy-to-use preparation routines for most common distributions after analyzing some real-life examples from customers.

The author’s assessment about the usefulness of this thesis work is extremely high, since it was successful in experimentally proving that DM works as an input verification method for an ERP system. It also enables to start building DM functionality inside Lean System ERP, but the future will show how this new functionality is implemented, if at all. Actually, during the final review with Development Manager Olli Veikkola from Tieto, it was discovered that the system already had some statistical checks implemented and, based on this, the conclusion about statistical methods was correct. Hopefully, at least their usage will be expanded in the future, since the demand for better data quality in the customer base is probably high even though customers do not necessarily know how the functionality to ensure that is implemented.

Critically thinking, the whole concept is still far from practical implementation, because there are so many open issues with the distribution transformations and ease of implementation, and the whole process for implementation is missing. Nevertheless, it is believed that this is worth serious future research, because the areas of DM application are increasing every day and the first company implementing a feasible solution will gain a huge competitive advantage. Apparently, the whole concept of using DM inside programs was given a name “predictive programming” by Mr. Lukawiecki and is being lectured around the world in, for example, Microsoft seminars (Lukawiecki, 2009a). This gives rise to expectations that this kind of functionality will become commonplace in the future.

There are diverse subjects for future work and research, like mapping out more problem regions and descriptive attributes with DM by exploring the data sets. Other interesting subjects might be algorithm and process optimization, a quantitative study about data quality degradation costs and verification of DM models against different customer distributions. And one should not forget the common applications of DM like tracing factory defects, exposing fraud and predicting sales. Some of these application targets might require that more descriptive attributes are stored in the database to facilitate better use of present data and, as a consequence, that could expand the applicability of ERP systems to completely new operational areas.

8 REFERENCES

Bayraktar, E., Jothishankar, M.C., Tatoglu, E. & Wu, T., 2007. Evolution of operations management: past, present and future. *Management Research News*, 30(11), pp.843-71.

Beyer, K., Goldstein, J., Ramakrishnan, R. & Shaft, U., 1999. When Is "Nearest Neighbor" Meaningful? In *Database Theory - ICDT '99*. Springer Berlin / Heidelberg. pp.217-35.

Boser, B.E., Guyon, I.M. & Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers. In *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*. Pittsburgh, Pennsylvania, US, 1992. ACM.

Brachman, R.J. & Anand, T., 1994. *The Process of Knowledge Discovery in Databases: A First Sketch*. Technical report. AAAI.

Breiman, L., Friedman, J.H., Olshen, R.A. & Stone, C.J., 1984. *Classification and Regression Trees*. Belmont, California, USA: Wadsworth Publishing Company.

Caro, A., Calero, C., Caballero, I. & Piattini, M., 2008. A proposal for a set of attributes relevant for Web portal data quality. *Software Quality Journal*, 16(4), pp.513-42.

Chapman, P. et al., 1999. *CRISP-DM Step-by-step data mining guide*. Industry standard. CRISP-DM consortium.

Eckerson, W.W., 2002. *Data quality and the bottom line: Achieving Business Success through a Commitment to High Quality Data*. The Data Warehousing Institute.

Ester, M., Kriegel, H.-P., Sander, J. & Xiaowei, X., 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proc. of 2nd International Conference on Knowledge Discovery and Data Mining (KDD '96)*., 1996. AAAI Press.

Fayyad, U., Haussler, D. & Stolorz, P., 1996a. KDD for Science Data Analysis: Issues and Examples. In *In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*. AAAI Press.

Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P., 1996b. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, 17, pp.37-54.

Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P., 1996c. Knowledge discovery and data mining: towards a unifying framework. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*. American Association for Artificial Intelligence.

Gartner, Inc., 2009. *Magic Quadrant for Midmarket and Tier 2-Oriented ERP for Product-Centric Companies*. [Online] Available at: http://mediaproducts.gartner.com/reprints/microsoft/vol4/article12/article12.html [Accessed 30 July 2009].

Glick, J., Buchanan, B. & Smith, R., 2008. *AITopics / Applications*. [Online] Available at: http://www.aaai.org/AITopics/pmwiki/pmwiki.php/AITopics/Applications [Accessed 24 July 2009]. Formerly American Association for Artificial Intelligence.

Granskog, B. & Veikkola, O., 2009. *Lean System data quality*. Interview. Tieto Finland Oy.

Grünwald, P.D., 2004. A Tutorial Introduction to the Minimum Description Length Principle. In Grünwald, P.D., Myung, J. & Pitt, M.A. *Advances in Minimum Description Length: Theory and Applications*. MIT Press. Ch. 1-2.

Hand, D., Mannila, H. & Smyth, P., 2001. *Principles of Data Mining*. The MIT Press.

Han, J. & Kamber, M., 2001. *Data Mining: Concepts and Techniques*. 1st ed. Morgan Kaufmann Publishers, Inc.

Hinneburg, A. & Keim, D.A., 1999. Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering. In *VLDB '99: Proceedings of the 25th International Conference on Very Large Data Bases*. San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

Hodge, V. & Austin, J., 2004. A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review*, 22(2), pp.85-126.

International Standards Office, 2008. *ISO/IEC 25012:2008 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) - Data quality model*. Geneva: ISO.

Jensen, D. & Cohen, R., 2000. Multiple Comparisons in Induction Algorithms. *Machine Learning*, 38(3), pp.309-38.

Kurgan, L.A. & Musilek, P., 2006. A survey of Knowledge Discovery and Data Mining process models. *The Knowledge Engineering Review*, 21(1), pp.1-24.

Lee, W. & Stolfo, S., 2000. A framework for constructing features and models for intrusion detection systems. *ACM Transactions on Information and System Security*, 3(4), pp.227-61.

Legendre, A.M., 1805. *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot.

Li, L. et al., 2004. Data mining techniques for cancer detection using serum proteomic profiling. *Artificial Intelligence in Medicine*, 32(2), pp.71-83.

Liu, H. & Yu, L., 2005. Toward integrating feature selection algorithms for classification and clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 17(4), pp.491-502.

Lloyd, S.P., 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28, pp.129-37.

Lukawiecki, R., 2008. *Introduction to Data Mining*. [Microsoft Technet presentation] Espoo: Microsoft (1.0) Available at: http://www.microsoft.com/emea/spotlight/sessionh.aspx?videoid=865 [Accessed 2 February 2008].

Lukawiecki, R., 2009a. Architecture of Predictive Programming. In *Microsoft TechEd Europe 2009*. Berlin Microsoft.

Lukawiecki, R., 2009b. Data Mining Process In-Depth. In *Microsoft TechEd Europe 2009*. Microsoft.

Madnick, S.E., Wang, R.Y., Yang, W.L. & Zhu, H., 2009. Overview and Framework for Data and Information Quality Research. *ACM Journal of Data and Information Quality*, 1(1), pp.1-22.

Maimon, O. & Rokach, L., eds., 2005. *The Data Mining and Knowledge Discovery Handbook*. Springer.

Maron, M.E., 1961. Automatic Indexing: An Experimental Inquiry. *J. ACM*, 8(3), pp.404-17.

Maron, M.E. & Kuhns, J.L., 1960. On Relevance, Probabilistic Indexing and Information Retrieval. *J. ACM*, 7(3), pp.216-44.

McCarthy, J., 2007. *Applications of AI*. [Online] Available at: http://www-formal.stanford.edu/jmc/whatisai/node3.html [Accessed 24 July 2009].

McQueen, J., 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*. Los Angeles, 1967. Universify of California Press.

Milenova, B.L. & Campos, M.M., 2002. O-Cluster: Scalable Clustering of Large High Dimensional Data Sets. In *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining*. Washington, DC, USA, 2002. IEEE Computer Society.

Milenova, B.L., Yarmus, J.S. & Campos, M.M., 2005. SVM in Oracle database 10g: removing the barriers to widespread adoption of support vector machines. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*. Trondheim, Norway, 2005. VLDB Endowment.

MIT, 2009. *The MIT Total Data Quality Management Program and The International Conference on Information Quality*. [Online] Available at: http://web.mit.edu/tdqm/www/index.shtml [Accessed 18 Nov 2009].

Nelder, J.A. & Wedderburn, R.W.M., 1972. Generalized Linear Models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3), pp.370-84.

Oracle, 2008a. *Oracle Data Mining Application Developer's Guide 11g Release 1 (11.1)*. Oracle.

Oracle, 2008b. *Oracle Database Mining Concepts 11g Release 1 (11.1).* [Online] Oracle Available at: http://www.oracle.com/pls/db111/to_pdf?pathname=datamine.111%2Fb28129.pdf&remark=portal+%28Books%29 [Accessed 1 Aug 2009].

Orr, K., 1998. Data quality and systems theory. *Communications of the ACM*, 41(2), pp.66-71.

Piatetsky-Shapiro, G., 1991. Knowledge Discovery in real databases: A report on the IJCAI-89 Workshop. *AI Magazine*, 11(5), pp.68-70.

Piatetsky-Shapiro, G., 2009. *Companies with Data Mining and Analytics Products.* [Online] Available at: http://www.kdnuggets.com/companies/products.html [Accessed 10 October 2009].

Pyle, D., 1999. *Data preparation for data mining.* San Francisco, CA, USA: Morgan Kaufmann Publisher Inc.

Raunio, R., 2010. *Lean System philosophy.* Interview. Tieto Finland Oy.

Redman, T.C., 1998. The impact of poor data quality on the typical enterprise. *Commun. ACM*, 41(2), pp.79-82.

Shankaranarayan, G., Ziad, M. & Wang, R.Y., 2003. Managing data quality in dynamic decision environments: An information product approach. *Journal of Database Management*, 14(4), pp.14-32.

Shtub, A., 2002. *Enterprise Resource Planning (ERP) : the dynamics of operations management.* Boston: Kluwer Academic.

Smola, A.J. & Schölkopf, B., 2004. A tutorial on support vector regression. *Statistics and Computing*, 14(3), pp.199-222.

Sumner, M., 1999. Critical success factors in enterprise wide information management systems projects. In Prasad, J., ed. *SIGCPR '99: Proceedings of the 1999 ACM SIGCPR Conference on Computer Personnel Research.* New Orleans, Louisiana, United States, 1999. ACM, New York.

Tan, P.-N., Steinbach, M. & Kumar, V., 2006. *Introduction to Data Mining.* Pearson Education, Inc.

Tieto Corporation Oyj, 2009. *Lean System.* [Online] Available at: http://www.tieto.com/leansystem [Accessed 28 February 2009].

Tieto Finland Oy, 2010. *Lean filosofia (Lean philosophy).* Philosophy presentation. Espoo, Finland.

Togaware Pty Ltd, 2010. *Togaware: Open Source Solutions and Data Mining.* [Online] Available at: http://www.togaware.com [Accessed 1 April 2010].

University of Ljubljana, 2010. *Orange.* [Online] Available at: http://www.ailab.si/orange [Accessed 5 April 2010].

University of Waikato, 2010. *Weka 3 - Data Mining with Open Source Machine Learning Software in Java*. [Online] Available at: http://www.cs.waikato.ac.nz/ml/weka/ [Accessed 6 April 2010].

Uusi-Rauva, E., Haverila, M. & Kouri, I., 1994. *Teollisuustalous*. 2nd ed. Infacs Johtamistekniikka Oy.

van Hillegersberg, J. & Kuldeep, K., 2000. ERP experiences and evolution. *Communications of the ACM*, April. pp.23-26.

Vapnik, V.N., 2000. *The Nature of Statistical Learning Theory*. 2nd ed. New York, USA: Springer-Verlag.

Veikkola, O., 2009. *Lean System capabilities*. Interview. Tieto Finland Oy.

Womack, J.P., Jones, D.T. & Roos, D., 1991. *The Machine That Changed the World : The Story of Lean Production*. Harper Perennial.

Xiaosong, Z. et al., 2008. The application study of ERP data quality assessment and improvement methodology. In *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on.*, 2008.

Xu, H., Nord, J.H., Brown, N. & Nord, G.D., 2002. Data quality issues in implementing an ERP. *Industrial Management & Data Systems*, 102(1), pp.47-58.

Yu, L. & Liu, H., 2004. Efficient Feature Selection via Analysis of Relevance and Redundancy. *The Journal of Machine Learning Research*, 5, pp.1205-24.

Yu, L., Ye, J. & Liu, H., 2007. *Teachings of Yu Lei*. [Online] Available at: http://www.cs.binghamton.edu/~lyu/SDM07/DR-SDM07.pdf [Accessed 7 Dec 2009].

Zhang, T., Ramakrishnan, R. & Livny, M., 1996. BIRCH: an efficient data clustering method for very large databases. *SIGMOD Rec.*, 25(2), pp.103-14.

Zhang, Z. & Sharifi, H., 2000. A Methodology for achieving agility in manufacturing organizations. *International Journal of Operations & Production Management*, 20(4), pp.496-513.